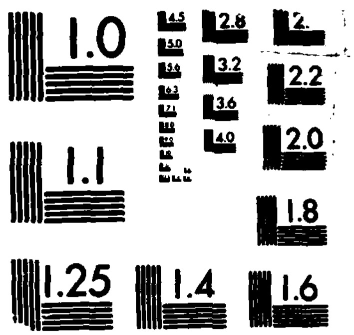
AD-A193 152 NUMERRCAL MODELING OF AIRBLAST(U) SCIENCE APPLICATIONS INTERNATIONAL CORP MCLEAN VA M A FRY JUN 87 SAIC-87/1701 N00014-86-C-2197 1/3

UNCLASSIFIED F/G 19/11 NN


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.8
3.2
3.6
4.0
4.5
5.0

(4)

NUMERICAL MODELING OF AIRBLAST

1ST YEAR FINAL REPORT

SAIC 87/1701

JUNE 1987

Science Applications International Corporation

4

NUMERICAL MODELING OF AIRBLAST

1ST YEAR FINAL REPORT

SAIC 87/1701

JUNE 1987

SAIC

Science Applications International Corporation

Post Office Box 1303, 1710 Goodridge Drive, McLean, Virginia 22102, (703) 821-4300

NUMERICAL MODELING OF AIRBLAST

1ST YEAR FINAL REPORT

SAIC 87/1701

June 1987

Submitted to:

Dr. Jay Boris
Laboratory for Computational Physics
Naval Research Laboratory
Washington, D.C. 20375

Prepared by:

Dr. Mark A. Fry

Prepared Under:

Contract No. N00014-86-C-2197

For the Period from March 9, 1986 to March 9, 1987

SCIENCE APPLICATIONS INTERNATIONAL CORPORATION

1710 GOODRIDGE DRIVE, MCLEAN, VIRGINIA 22102
(703) 734-5840

-A-

TABLE OF CONTENTS:

NUMERICAL MODELING OF AIRBLAST

CONTRACT No. N00014-86-C-2197

1st Year Final Report

Introduction

The SAIC effort during the period March 10, 1986 to March 10, 1987 is described. Included in this report are appendices for the deliverables such as fortran code listings. The effort was scaled back due to budget considerations. The areas that are contained herein are Tasks 2.1, 2.2, 2.4, 2.5, 2.6, 2.7, 2.8, and 2.9 as described in the statement of work. SAIC over the last year has provided a highly qualified staff of research scientists to perform the tasks described under the contract statement of work. Dr. Mark A. Fry has acted as principal scientist for the contract work. Dr. Shmuel Eidelman has worked on site for 8 months and then 4 months offsite at SAIC. Mr. Ken Laskey assisted as a junior level research scientist for 3 months. Mr. Mike Zuniga joined the effort in November 1986 as a junior level research scientist.

TASK 2.1

The 2-D axisymetric FCT code was applied to nuclear and non-nuclear problems. A burn routine for HE has been added to the code but has not been thoroughly checked. The Minor-scale event was simulated and waveforms at various station locations were obtained. These were used in a comparison with a full 3-D simulation and with the experimental data it was determined that the simulated thermal layer produced multiple peaks in the overpressure waveforms. This was primarily due to a tear in the plastic bag used to contain the helium. The numerical simulations did, however, agree quite well. The results of this work were presented at the Minor Scale Symposium in Albuquerque.

The 2-D code was further optimized for running on the CRAY class computers by employing pointers in the data structure. Speed increases of 10 to 15% were obtained.

Work was also performed in addressing improvements for data management in 2-D codes. This work has been titled "Monotonic Logical Grid Algorithms". Mr. Mike Zuniga has been working full time on these improvements.

A simulation program has been written to test the algorithm. In addition, a graphics program has also been written to aid in this research.

TASK 2.2

A new first principles model for dust ingestion was developed. The momentum integral equation is solved in 1-D along the ground surface. This equation along with boundary layer physical assumptions provides an unsteady prediction of the mass flux emerging from the ground. This model was first proposed by Mirels and is now implemented into the 2-D FCT code. The option DUST=2 initiates the use of this model. A listing of the 2-D code and the subroutine, DUSTN, can be found in Appendix A.

TASK 2.4

The 3-D FCT code was used to study the late time cloud rise geometry from multiple nulear explosions. Calculations involving two bursts separated by distance and time were executed. Cases of 1 MT bursts at 20 and 30 second separation with distance separation of 250 m, 500 m, 1 km, and 3 km were completed. The results indicate a strong sensitivity in cloud formation with a small distance

separation. Previous system modeling codes assumed no sensitivity and have now been corrected.

Spike attacks involving many bursts simultaneously in a 20 x 5 km grid were investigated. Simulations performed with the 3-D code indicated a strong asymmetry out to several minutes in time and up to 40 to 50 km in altitude. A paper summarizing the results was presented at the DNA DUST Symposium in July , 1986. The abstract of the paper is presented in Appendix B. Additional presentations were made at DNA headquarters.

As an additional item, particles were added to the 3-D code for tracking material with time. The 3-D code was used to study the possiblity of using distributed chemical charges in array geometries. Numerous chemical energy release models where employed. The results are summarized in a report in Appendix C.

TASK 2.5

Work on this task was not performed due to budget limitations.

TASK 2.6

Work on this task was not performed due to budget limitations.

TASK 2.7

Work on this task was not performed due to budget limitations.

TASK 2.8

The 3-D graphics program has been updated and improved for hard copy display on Tektronix compatible terminals. Links were written to use DISSPLA as well as the NCAR graphics package. A choice of bold face lettering is now also available. The extensive changes and the improved documentation are available in the source listing, Appendix D .

TASK 2.9

Task 2.9 was not performed due to budget considerations.

APPENDIX A

SOURCE LISTING OF THE 2-D AXISYMMETRIC AND CARTESIAN NUCLEAR AND NON-NUCLEAR MODELLING CODE

```
      program sp2r2d
c----------------------------------------------------------------
c     a 2d fct code for high explosive and nuclear blasts in air.
c     coded for cartesian (x-y) and cylindrical (r-z)
c     coordinate systems.
c----------------------------------------------------------------
      parameter (mnx=150, mny=200, mxy=202)
      parameter (mnxx=1, mnyy=1)
      parameter (nsta=1)
      parameter (nop=1)
      parameter (mxx=1)
c
      integer alpha
      logical lmgrid, lstat, lnuc, therml, lgrav, leos
      real rha(mnx,mny), rhb(mnxx,mnyy), rhc(mnxx,mnyy)
      real tem(mnxx,mnyy), scl(mnxx,mnyy), rho(mnx,mny)
      real rvx(mnx,mny), rvy(mnx,mny), erg(mnx,mny)
      common nx, ny, nxp, nyp, nspec, idump, idiag, lmgrid
      common lstat, lnuc, therml, lgrav, leos
      common alpha, gamma, gm1, dt, dx, dy, rhomin
      common tem, scl, rho, rvx, rvy, erg, rha, rhb, rhc
c
      real xcor(mxy), xcoro(mxy), ycor(mxy), ycoro(mxy), unit(mxy)
      common /grids/ xcor, xcoro, ycor, ycoro, unit
c
      real xbnd(5), dxr(4), ybnd(3), dyr(2)
      common /grdcon/ ndx, nsmth, mdy, alt, nahed, xfine, hx, hy,
     1 xbnd, ybnd, dxr, dyr, xleft
c
      real rhom(mxy), pres(mxy), vel(mxy), uh(mxy)
      real ergk(mxy), srvx(mxy), srvy(mxy), serg(mxy)
      real rha0(mxy), rhb0(mxy), rhc0(mxy), vh(mxy)
      real rho0(mxy), rvx0(mxy), rvy0(mxy), erg0(mxy)
      real rhoh(mxy), rvxh(mxy), rvyh(mxy), ergh(mxy)
      real delx(mxy), dely(mxy), scrh(mxy), sgrv(mxy), sgam(mxy)
      real rmin(mxy), rmax(mxy), rvr(mxy)
      common /scrch/ rhom, pres, vel, uh, ergk, srvx, srvy, serg,
     1 rha0, rhb0, rhc0, vh, rho0, rvx0, rvy0, erg0, rhoh, rvxh, rvyh
     2 , ergh, delx, dely, scrh, sgrv, sgam, rmin, rmax, rvr
c
      logical part
      real xp(nop), yp(nop)
      common /part/ part, xp, yp, nopp
c
      real rhs(mxx,nsta), vis(mxx,nsta), gms(mxx,nsta), tme(mxx)
      real prs(mxx,nsta), xs(nsta), ys(nsta)
      common /stat/ rhs, vis, gms, prs, tme, xs, ys, fracn, eblast,
     1 ixx
c
      common /tstep/ dtmin, dtmax
      common /jwl/ r0cj, pcj, dcj, e0cj, gcj, acj, beos, ccj, r1cj,
     1 r2cj, wcj
c
      real pr00(mxy), rh00(mxy), er00(mxy), gp00(mxy), gravy(mxy)
      parameter (jthm=1)
      real fsky(mxy), rhtl(mnx,jthm)
      common /grav/ pr00, rh00, er00, gp00, gravy, fsky, rhtl
     1 , jtherm
c
      logical lscan, rigt, left, ltop, botm
      common /scan/ lscan, rigt, left, ltop, botm
      common /active/ ixr, jxr, ixl, jxl, iyt, jyt, iyb, jyb, iir,
     1 jbt, iadd, jadd, factor
c
```

```
      integer itape, otape, ifile, ofile
      logical rezone, irest, isave, igrid, dust, idump, imgold
     x        , dustn
      character *8 prntf, restf, dumpf, readf
      character *25 path
      character *40 glab
      character *40 label
      common /inout/ itape, otape, ifile, ofile, isave, npath, nlab
c
      parameter (ntdump=57)
      real dmptim(ntdump)
      logical ltdump
      data idump /.true./
c
      data readf /'megdat'/, prntf /'megprt'/, dumpf /'sp2r'/
      data minstp, maxstp /1,11/
      data itape, otape, ifile, ofile /14,15,0,1/
c
      data ndx, nsmth, mdy /75,25,37/
      data hx, hy /1.0,1.0/
      data xfine, alt, xleft /0.0,0.0,0.0/
      data nahed /30/
      data ixx /0/
c
      data dtmin, dtmax, dtstrt /1.0e-6,2.0e-5,1.e-5/
      data jtherm /0/, nbod /0/
c
      namelist /contrl/ minstp,maxstp,idiag,idump,imgrid,leos,lnuc
     1 ,itape,otape,ifile,ofile,lstat,part,therml,igrav,irest,isave
     2 ,ltdump,dmptim
      namelist /mesh/ nx,ny,alpha,dx,dy,ndx,nsmth,mdy,xfine,alt
     1 ,nahed,hx,hy,iadd,jadd,factor,lscan,rigt,left,ltop,botm,igrid
     2 ,xleft
      namelist /bcond/ bcr,bcl,bct,bcb
      namelist /phydat/ gamma,rhomin,dtmin,dtmax,nspec,jtherm,jdirt
     1 ,dtstrt,dust,nbod,dustn
      namelist /jwlist/ r0cj,pcj,dcj,e0cj,gcj,acj,beos,ccj,r1cj,r2cj
     1 ,wcj
      namelist /stats/ xs,ys
      namelist /newgrd/ ixr,ixl,jyt,jyb
c
c    initialize the code control parameters.
      idiag=10000
      idump=10000
      imgrid=.false.
      lstat=.false.
      part=.false.
      leos=.false.
      lnuc=.false.
      igrav=.false.
      therml=.false.
      rezone=.false.
      lscan=.false.
      rigt=.false.
      left=.false.
      ltop=.false.
      botm=.false.
      isave=.false.
      igrid=.false.
      dust=.false.
      dustn=.false.
      ltdump=.false.
c
c    initialize grid parameters.
```

```
      nx=mnx
      ny=mny
      nxp=nx+1
      nyp=ny+1
      alpha=1
      dx=1.0
      dy=1.0
      ixr=mnx
      jxr=1
      ixl=1
      jxl=1
      iyt=1
      jyt=mny
      iyb=1
      jyb=1
      ilr=mnx
      jbt=mny
      iadd=10
      jadd=10
c
c     initialize boundary conditions.
      bcr=0.0
      bcl=0.0
      bct=0.0
      bcb=-1.0
c
c     physical data.
      rhomin=1.29e-06
      gamma=1.4
      gm1=gamma-1.0
      nspec=1
c
      time=0.0
      tme(1)=time
      radl=0.0
c
c     open read and print files. read and write data.
      call bufman
      open (unit=5,file=readf,status='old')
      read (5,180) label
      read (5,200) prntf
      read (5,200) dumpf
      read (5,200) restf
      read (5,180) path
      nlab=0
      do 10 i=1,40
      nlab=nlab+1
      if (label(i:i).eq.'$') go to 20
   10 continue
   20 npath=0
      do 30 i=1,25
      if (path(i:i).eq.' ') go to 40
   30 npath=npath+1
   40 continue
c
      read (5,contrl)
      if (.not.lrest) then
      do 50 i=8,1,-1
      prntf(i:i)=dumpf(i:i)
      if (prntf(i:i).ne.' '.and.prntf(i+1:i+1).eq.' ') prntf(i+1:i+1
     1 )='0'
   50 continue
      ifile=0
      ofile=0
```

```
      endif
      open (unit=6,file=prntf,status='new')
      write (6,190) label
      write (6,contrl)
      read (5,mesh)
      write (6,mesh)
      read (5,bcond)
      write (6,bcond)
      read (5,phydat)
      write (6,phydat)
      dt=dtstrt
      if (nspec.gt.1) read (5,jwlist)
      if (nspec.gt.1) write (6,jwlist)
      if (.not.therml) jtherm=0
      if (.not.lstat) go to 70
      read (5,stats)
      write (6,stats)
      do 60 ksta=1,nsta
   60 ys(ksta)=ys(ksta)+alt
      if (idump.le.mxx) go to 70
      write (6,220) idump,mxx
      go to 160
   70 continue
      do 80 k=1,mxy
   80 unit(k)=1.0
c
c     determine bluff body profiles.
      if (nbod.ne.0) call bluf (nbod)
c
c     initialize variables to ambient conditions.
      if (.not.lrest) then
      call inital (bcl,bcr,bcb,bct)
      if (dust) call dust0
      if(dustn) call dust01
      if (therml) call tlayer (rho,rvx,rvy,erg,rhomin,ensum)
c      if (lnuc) call dposit (time)
      if (lnuc) then
      call burst (xcor,ycor,erg)
c      call dposit (time)
      else
      call jump (gamma,leos,er00(2))
      call dpshok (rho,rvx,rvy,erg)
      endif
      call dumper (0,time,dumpf,label,path,glab)
      call diagno (istep,time)
      go to 160
      endif
c
c     restart if lrest=.true.
      lmgold=lmgrid
      call restrt (minstp,time,restf,glab)
      lmgrid=lmgold
      if (.not.lnuc) call jump (gamma,leos,er00(2))
c
c     determine dump time number.
      itdump=1
      if (ltdump) then
      do 90 ktdump=ntdump,1,-1
      if (time.ge.dmptim(ktdump)) go to 100
   90 continue
      go to 110
  100 itdump=ktdump+1
      ltdump=itdump.le.ntdump
  110 continue
```

```
        endif
c
c     read in boundaries of active grid.
        if (lgrid) then
        read (5,newgrd)
        write (6,newgrd)
        ilr=ixr-ixl+1
        jbt=jyt-jyb+1
        endif
c
c     initialize various routines.
       ratio=rho(mnx,7)/rho(mnx,1)
       write(6,987) ratio
 987   format(1x,'ratio=',1pe12.5,' rad loss fbloss')
        if (lnuc) call fbloss (time,dt,radl,eblast,ratio)
        if (lscan) call xyg
        iactiv=min0(iadd,jadd)
c
c     begin the loop over time steps.
        do 150 istep=minstp,maxstp
        kstep=istep-minstp
c
c     determine active grid boundaries.
        if (lscan) then
        if (mod(kstep,iactiv).eq.0) call xygrid (erg,scrh)
        endif
c
c     the time and the timestep are updated.
        call dtset (istep,time,radl,ensum)
        if (nbod.ne.0) go to 130
c
c     update grid.
        if (.not.lmgrid) go to 120
        call xgrid (jtherm)
        call ygrid
  120   continue
c
c     integrate along rows and columns.
        call bndrx (bcr,bcl)
        call integx
        call bndry (bct,bcb)
        call integy
        go to 140
c
c     integration for shock on bluff bodies.
  130   call blufxy (bcl,bcr,bcb,bct)
  140   continue
c
c     perform radiation loss calculation.
        if (lnuc) call radlos (time,xcor,ycor,radl,fracn)
c
c     calculate station data.
        if (lstat) call staton (ixx,istep,time)
c
c     push particles
        if (part) call partic (xcor,ycor)
c
c     inject dust.
        if (dust) call duster(istep,time)
        if(dustn) call dustm(istep,time)
c
c     dump at the appropriate step.
        if (ltdump) then
        ldump=(istep.eq.maxstp).or.(time.ge.dmptim(itdump))
```

```
      else
      ldump=(istep.eq.maxstp).or.(idump.ne.0.and.mod(istep,idump).eq
     1 .0)
      endif
      ldump=cvmgt(.false.,ldump,istep.eq.minstp)
      if (ldump) then
      call flush
      call dumper (istep,time,dumpf,label,path,glab)
      call flush
      ixx=0
      if (ltdump) then
      if (time.ge.dmptim(ntdump)) go to 160
      itdump=itdump+1
      ltdump=itdump.le.ntdump
      endif
      endif
c
c     perform all required diagnostics
      if (idiag.ne.0.and.mod(istep,idiag).eq.0) call diagno (istep
     1 ,time)
      if(time.gt.25.) go to 170
  150 continue
c
c     save print file.
  160 write (6,210) ptrem
      if (.not.lsave) go to 170
      write (glab,230) path(1:npath),prntf
      ierr=1000
      call mass ('save','\wait=on\',0,glab,ierr)
      write (6,240) prntf,ierr
c
c     close files.
  170 close (unit=5,status='keep')
      close (unit=6,status='keep')
      stop
c
  180 format (a)
  190 format ('1fast2d hydrodynamics  ',5x,a)
  200 format (15x,a)
  210 format ('0ptrem = ',e13.5)
  220 format (1x,'idump,mxx=',2i5/1x,'idump should be .le. mxx')
  230 format ('&',a,a,'&')
  240 format (1x,'ierr for ',a,' =',i4/)
      end
      subroutine xygrid (erg,scr1)
c-----------------------------------------------------------------
c     determines extent of active grid.
c-----------------------------------------------------------------
      parameter (mnx=150, mny=200, mxy=202)
      real erg(mnx,mny), scr1(mxy), scr2(mxy)
c
      real pr00(mxy), rh00(mxy), er00(mxy), gp00(mxy), gravy(mxy)
      parameter (jthm=1)
      real fsky(mxy), rhtl(mnx,jthm)
      common /grav/ pr00, rh00, er00, gp00, gravy, fsky, rhtl
     1 , jtherm
c
      logical lscan, rigt, ltop, left, botm
      common /scan/ lscan, rigt, left, ltop, botm
      common /active/ ixr, jxr, ixl, jxl, iyt, jyt, iyb, jyb, ilr,
     1 jbt, iadd, jadd, factor
c
c     save old grid boundaries.
      ixro=ixr
```

```
      ixlo=ixl
      jyto=jyt
      jybo=jyb
c
c   right.
      if (.not.rigt) go to 30
      ixr=1
      do 20 j=jmin,jmax
      do 10 i=imin,imax
10    scr1(i)=cvmgt(float(i),0.0,erg(i,j).ge.scr2(j))
      klim=ismax((imax-imin+1),scr1(imin),1)+imin-1
      ixr=max0(klim,ixr)
      jxr=cvmgt(j,jxr,ixr.eq.klim)
20    continue
      ixr=min0((ixr+iadd),mnx)
      imax=ixr
      rigt=cvmgt(.false.,.true.,ixr.eq.mnx)
c
c   top.
30    if (.not.ltop) go to 60
      jyt=1
      do 50 i=imin,imax
      do 40 j=jmin,jmax
40    scr1(j)=cvmgt(float(j),0.0,erg(i,j).ge.scr2(j))
      klim=ismax((jmax-jmin+1),scr1(jmin),1)+jmin-1
      jyt=max0(klim,jyt)
      iyt=cvmgt(i,iyt,jyt.eq.klim)
50    continue
      jyt=min0((jyt+jadd),mny)
      jmax=jyt
      ltop=cvmgt(.false.,.true.,jyt.eq.mny)
c
c   left.
60    if (.not.left) go to 90
      ixl=mnx
      do 80 j=jmin,jmax
      do 70 i=imin,imax
70    scr1(j)=cvmgt(float(i),float(mnx),erg(i,j).ge.scr2(j))
      klim=ismin((imax-imin+1),scr1(imin),1)+imin-1
      ixl=cvmgt(min0(klim,ixl),ixl,klim.ne.imin)
      jxl=cvmgt(j,jxl,ixl.eq.klim)
80    continue
      ixl=max0((ixl-iadd),1)
      imin=ixl
      left=cvmgt(.false.,.true.,ixl.eq.1)
c
c   bottom.
90    if (.not.botm) go to 120
      jyb=mny
      do 110 i=imin,imax
      do 100 j=jmin,jmax
100   scr1(j)=cvmgt(float(j),float(mny),erg(i,j).ge.scr2(j))
      klim=ismin((jmax-jmin+1),scr1(jmin),1)+jmin-1
      jyb=cvmgt(min0(klim,jyb),jyb,klim.ne.jmin)
      iyb=cvmgt(i,iyb,jyb.eq.klim)
110   continue
      jyb=max0((jyb-jadd),1)
      jmin=max0(jyb,jtherm+1)
      botm=cvmgt(.false.,.true.,jyb.eq.1)
120   ixr=cvmgt(max0(ixr,ixro),ixr,ixro.ne.mnx)
      ixl=cvmgt(min0(ixl,ixlo),ixl,ixlo.ne.1)
      jyt=cvmgt(max0(jyt,jyto),jyt,jyto.ne.mny)
      jyb=cvmgt(min0(jyb,jybo),jyb,jybo.ne.1)
      ilr=ixr-ixl+1
```

```
      jbt=jyt-jyb+1
      lscan=cvmgt(.false.,.true.,ixl.eq.1.and.ixr.eq.mnx.and.jyb.eq.
     1 1.and.jyt.eq.mny)
      return
c
      entry xyg
      imin=ixl
      imax=ixr
      jmin=max0(jyb,jtherm+1)
      jmax=jyt
      do 130 j=1,mny
 130  scr2(j)=factor*er00(j+1)
      return
      end
      subroutine integx
c-----------------------------------------------------------------
c     integrates along rows.
c--------------------------------------------------------------------
      parameter (mnx=150, mny=200, mxy=202)
      parameter (mnxx=1, mnyy=1)
      integer alpha
      logical lmgrid, lstat, lnuc, therml, lgrav, leos
      real rha(mnx,mny), rhb(mnxx,mnyy), rhc(mnxx,mnyy)
      real tem(mnxx,mnyy), scl(mnxx,mnyy), rho(mnx,mny)
      real rvx(mnx,mny), rvy(mnx,mny), erg(mnx,mny)
      common nx, ny, nxp, nyp, nspec, idump, idiag, lmgrid
      common lstat, lnuc, therml, lgrav, leos
      common alpha, gamma, gm1, dt, dx, dy, rhomin
      common tem, scl, rho, rvx, rvy, erg, rha, rhb, rhc
c
      real xcor(mxy), xcoro(mxy), ycor(mxy), ycoro(mxy), unit(mxy)
      common /grids/ xcor, xcoro, ycor, ycoro, unit
c
      real rhom(mxy), pres(mxy), vel(mxy), uh(mxy)
      real ergk(mxy), srvx(mxy), srvy(mxy), serg(mxy)
      real rha0(mxy), rhb0(mxy), rhc0(mxy), vh(mxy)
      real rho0(mxy), rvx0(mxy), rvy0(mxy), erg0(mxy)
      real rhoh(mxy), rvxh(mxy), rvyh(mxy), ergh(mxy)
      real delx(mxy), dely(mxy), scrh(mxy), sgrv(mxy), sgam(mxy)
      real rmin(mxy), rmax(mxy), rvr(mxy)
      common /scrch/ rhom, pres, vel, uh, ergk, srvx, srvy, serg,
     1 rha0, rhb0, rhc0, vh, rho0, rvx0, rvy0, erg0, rhoh, rvxh, rvyh
     2 , ergh, delx, dely, scrh, sgrv, sgam, rmin, rmax, rvr
c
      real rscr(mxy), pscr(mxy), gscr(mxy), scr1(mxy), scr2(mxy)
      equivalence (rscr,srvx), (pscr,srvy), (gscr,serg)
      equivalence (scr1,scrh), (scr2,rmin)
      real gams(mxy), rhah(mxy), rhbh(mxy), rhch(mxy), rgrv(mxy)
      equivalence (gams,sgam), (rgrv,dely)
      equivalence (rhah,rmax), (rhbh,rvr), (rhch,delx)
c
      real pr00(mxy), rh00(mxy), er00(mxy), gp00(mxy), gravy(mxy)
      parameter (jthm=1)
      real fsky(mxy), rhtl(mnx,jthm)
      common /grav/ pr00, rh00, er00, gp00, gravy, fsky, rhtl
     1 , jtherm
c
      common /active/ ixr, jxr, ixl, jxl, iyt, jyt, iyb, jyb, ilr,
     1 jbt, iadd, jadd, factor
c
      common /lbc/ rholbc, rvxlbc, rvylbc, erglbc
      common /rbc/ rhorbc, rvxrbc, rvyrbc, ergrbc
      common /bbc/ rhobbc, rvxbbc, rvybbc, ergbbc
      common /tbc/ rhotbc, rvxtbc, rvytbc, ergtbc
```

```
c
      call ngridd (xcoro(ixl),iir+1,alpha)
      call ogridd (iir+1)
      call ngridd (xcor(ixl),iir+1,alpha)
c
c     loop over rows to be integrated.
      do 250 j=jyb,jyt
      rhor=cvmgt(rhoO(j+1),rhor,abcr.eq.1.0)
      rhol=cvmgt(rhoO(j+1),rhol,abcl.eq.1.0)
      ergr=cvmgt(er00(j+1),ergr,abcr.eq.1.0)
      ergl=cvmgt(er00(j+1),ergl,abcl.eq.1.0)
      rhcr=rhor
      rhcl=rhol
      do 10 i=ixl,ixr
      rho0(i)=rho(i,j)
      rhc0(i)=rho(i,j)
      rha0(i)=0.0
      rhb0(i)=0.0
      rvx0(i)=rvx(i,j)
      rvy0(i)=rvy(i,j)
      erg0(i)=erg(i,j)
      rhoh(i)=rho0(i)
      rvxh(i)=rvx0(i)
      rvyh(i)=rvy0(i)
   10 ergh(i)=erg0(i)
c
      go to (60,40,20), nspec
   20 do 30 i=ixl,ixr
      rhb0(i)=rhb(i,j)
      rhbh(i)=rhb0(i)
   30 rhc0(i)=rhc0(i)-rhb0(i)
      rhcr=rhcr-rhbr
      rhcl=rhcl-rhbl
   40 do 50 i=ixl,ixr
      rha0(i)=rha(i,j)
      rhah(i)=rha0(i)
      rhc0(i)=amax1((rhc0(i)-rha0(i)),0.0)
   50 rhch(i)=rhc0(i)
      rhcr=rhcr-rhar
      rhcl=rhcl-rhal
   60 continue
c
c     define boundary velocities by averaging values in interior and
c     exterior boundary cells.
      rvxlef=frvxl*rvx(ixl,j)+rvxl
      rholef=frhol*rho(ixl,j)+rhol
      uh(ixl)=rvx(ixl,j)/rho(ixl,j)
      uh(ixl)=0.5*(uh(ixl)+rvxlef/rholef)
c
      rvxrig=frvxr*rvx(ixr,j)+rvxr
      rhorig=frhor*rho(ixr,j)+rhor
      uh(ixr+1)=rvx(ixr,j)/rho(ixr,j)
      uh(ixr+1)=0.5*(uh(ixr+1)+rvxrig/rhorig)
c
c     advance total density, energy and x and y momentum for
c     half and then whole step.
      do 190 ipass=1,2
      dtsub=dt/float(3-ipass)
c
c     calculate transport velocities.
      do 70 i=ixl,ixr
      rhom(i)=1.0/rhoh(i)
      vel(i)=rvxh(i)*rhom(i)
   70 ergk(i)=amax1(0.0,ergh(i)-0.5*rhom(i)*(rvxh(i)*rvxh(i)+rvyh(i)
```

```
     1    *rvyh(i))-rhoh(i)*gp00(j+1))
          do 80 i=ixl+1,ixr
   80     uh(i)=0.5*(vel(i)+vel(i-1))
          call velocd (uh(ixl),(ilr+1),dtsub,ilr)
c
c     calculate pressure.
          if (leos) then
c          call eos (ilr,nspec,rhoh(ixl),ergk(ixl),gams(ixl),pres(ixl)
c    1      ,rscr(ixl),pscr(ixl),gscr(ixl),scr1(ixl),scr2(ixl),rhah(ixl)
c    2      ,rhoh(ixl))
          call eospl (rhoh(ixl),ergk(ixl),ilr,gams(ixl),pres(ixl))
          else
          do 90 i=ixl,ixr
   90     pres(i)=gm1*ergk(i)
          endif
c
c     calculate source terms.
          do 100 i=ixl,ixr
          srvx(i)=-pres(i)
  100     serg(i)=srvx(i)*vel(i)
          srvxl=-pres(ixl)
          srvxr=-pres(ixr)
          sergl=-pres(ixl)*uh(ixl)
          sergr=-pres(ixr)*uh(ixr+1)
c
c     advance total density, energy and x and y momentum.
          if (abs(abcl).eq.1.0) call setbd (1,1)
          call fryfct (rho0(ixl),rhoh(ixl),ilr,frhor,rhor,frhol,rhol)
          do 110 i=ixl,ixr
  110     rhoh(i)=amax1(rhoh(i),rhomin)
          call sourcd (ilr,dtsub,2,unit(ixl),srvx(ixl),srvxr,srvxl)
          call fryfct (rvx0(ixl),rvxh(ixl),ilr,frvxr,rvxr,frvxl,rvxl)
          call fryfct (rvy0(ixl),rvyh(ixl),ilr,frvyr,rvyr,frvyl,rvyl)
          call sourcd (ilr,dtsub,1,unit(ixl),serg(ixl),sergr,sergl)
          call fryfct (erg0(ixl),ergh(ixl),ilr,fergr,ergr,fergl,ergl)
c
c     advance species.
          do 120 i=ixl,ixr
          scr1(i)=1.e-8*rhoh(i)
  120     scr2(i)=0.0
          go to (190,150,130), nspec
  130     call fryfct (rhb0(ixl),rhbh(ixl),ilr,frhbr,rhbr,frhbl,rhbl)
          do 140 i=ixl,ixr
          rhbh(i)=cvmgt(0.0,rhbh(i),rhbh(i).le.scr1(i))
  140     scr2(i)=scr2(i)+rhbh(i)
  150     call fryfct (rha0(ixl),rhah(ixl),ilr,frhar,rhar,frhal,rhal)
          do 160 i=ixl,ixr
          rhah(i)=cvmgt(0.0,rhah(i),rhah(i).le.scr1(i))
  160     scr2(i)=scr2(i)+rhah(i)
          call fryfct (rhc0(ixl),rhch(ixl),ilr,frhcr,rhcr,frhcl,rhcl)
          do 170 i=ixl,ixr
          rhch(i)=cvmgt(0.0,rhch(i),rhch(i).le.scr1(i))
  170     scr2(i)=scr2(i)+rhch(i)
          do 180 i=ixl,ixr
  180     scrh(i)=rhoh(i)/scr2(i)
  190     continue
c
          do 200 i=ixl,ixr
          rho(i,j)=rhoh(i)
          rvx(i,j)=rvxh(i)
          rvy(i,j)=rvyh(i)
  200     erg(i,j)=ergh(i)
          go to (250,230,210), nspec
  210     do 220 i=ixl,ixr
```

```
  220     rhb(i,j)=rhbh(i)*scrh(i)
  230     do 240 i=ixl,ixr
  240     rha(i,j)=rhah(i)*scrh(i)
c
c      loop back for the next row to be integrated.
  250     continue
          return
c
          entry integy
c ------------------------------------------------------------------
c      integrates along columns.
c ------------------------------------------------------------------
          call ngridd (ycoro(jyb),jbt+1,1)
          call ogridd (jbt+1)
          call ngridd (ycor(jyb),jbt+1,1)
c
c      loop over the cols to be integrated.
          do 520 i=ixl,ixr
          do 260 j=jyb,jyt
          rho0(j)=rho(i,j)
          rhc0(j)=rho(i,j)
          rha0(j)=0.0
          rhb0(j)=0.0
          rvx0(j)=rvx(i,j)
          rvy0(j)=rvy(i,j)
          erg0(j)=erg(i,j)
          rhoh(j)=rho0(j)
          rvxh(j)=rvx0(j)
          rvyh(j)=rvy0(j)
  260     ergh(j)=erg0(j)
c
          go to (310,290,270), nspec
  270     do 280 j=jyb,jyt
          rhb0(j)=rhb(i,j)
          rhbh(j)=rhb0(j)
  280     rhc0(j)=rhc0(j)-rhb0(j)
          rhct=rhct-rhbt
          rhcb=rhcb-rhbb
  290     do 300 j=jyb,jyt
          rha0(j)=rha(i,j)
          rhah(j)=rha0(j)
          rhc0(j)=amax1((rhc0(j)-rha0(j)),0.0)
  300     rhch(j)=rhc0(j)
          rhct=rhct-rhat
          rhcb=rhcb-rhab
  310     continue
c
c      set thermal layer densities.
          do 312 j=1,mxy
  312     rgrv(j)=rh00(j)
          if (thermal) then
          do 314 j=1,jthm
  314     rgrv(j+1)=rht1(i,j)
          endif
c
c      define boundary velocities by averaging values in interior and
c      exterior boundary cells.
          rvybot=frvyb*rvy(i,jyb)+rvyb
          rhobot=frhob*rho(i,jyb)+rhob
          vh(jyb)=rvy(i,jyb)/rho(i,jyb)
          vh(jyb)=0.5*(vh(j,b)+rvybot/rhobot)
c
          rvytop=frvyt*rvy(i,jyt)+rvyt
          rhotop=frhot*rho(i,jyt)+rhot
```

```
      vh(jyt+1)=rvy(i,jyt)/rho(i,jyt)
      vh(jyt+1)=0.5*(vh(jyt+1)+rvytop/rhotop)
c
c     advance total density, energy and x and y momentum
c     for half and then whole step.
      do 460 ipass=1,2
      dtsub=dt/float(3-ipass)
c
c     calculate transport velocities.
      do 320 j=jyb,jyt
      rhom(j)=1.0/rhoh(j)
      vel(j)=rvyh(j)*rhom(j)
  320 ergk(j)=amax1(0.0,ergh(j)-0.5*rhom(j)*(rvxh(j)*rvxh(j)+rvyh(j)
     1 *rvyh(j))-rhoh(j)*gp00(j+1))
      do 330 j=jyb+1,jyt
  330 vh(j)=0.5*(vel(j)+vel(j-1))
      call velocd (vh(jyb),(jbt+1),dtsub,jbt)
c
c     calculate pressure.
      if (leos) then
c      call eos (jbt,nspec,rhoh(jyb),ergk(jyb),gams(jyb),pres(jyb)
c    1  ,rscr(jyb),pscr(jyb),gscr(jyb),scr1(jyb),scr2(jyb),rhah(jyb)
c    2  ,rhbh(jyb))
      call eospi (rhoh(jyb),ergk(jyb),jbt,gams(jyb),pres(jyb))
      else
      do 340 j=jyb,jyt
  340 pres(j)=gm1*ergk(j)
      endif
c
c     calculate source terms.
      if (lgrav) then
      do 350 j=jyb,jyt
c      srvy(j)=pr00(j+1)-pres(j)
c 350  sgrv(j)=-(rgrv(j+1)-rhoh(j))*gravy(j+1)
c      srvyb=pr00(jyb)-pres(jyb)
c      srvyt=pr00(jyt+1)-pres(jyt)
c      sgrv(jyt+1)=-(rgrv(jyt+2)-rhoh(jyt))*gravy(jyt+2)
      srvy(j)=pr00(j+1)-pres(j)-fsky(j)
  350 sgrv(j)=rhoh(j)*gravy(j+1)
      srvyb=pr00(jyb)-pres(jyb)-fsky(jyb)
      srvyt=pr00(jyt+1)-pres(jyt)-fsky(jyt+1)
      sgrv(jyt+1)=rhoh(jyt)*gravy(jyt+2)
      else
      do 355 j=jyb,jyt
  355 srvy(j)=-pres(j)
      srvyb=-pres(jyb)
      srvyt=-pres(jyt)
      endif
      do 370 j=jyb,jyt
c 370  serg(j)=-pres(j)*vel(j)
c      sergt=-pres(jyt)*vh(jyt+1)
c      sergb=-pres(jyb)*vh(jyb)
  370 serg(j)=-(pres(j)+fsky(j+1)-pr00(j+1))*vel(j)
      sergt=-(pres(jyt)+fsky(jyt+2)-pr00(jyt+2))*vel(jyt+1)
      sergb=-(pres(jyb)+fsky(jyb)-pr00(jyb))*vh(jyb)
c
c     advance density, energy and x and y momentum.
      if (abs(abcb).eq.1.0) call setbd (1,1)
      call fryfct (rho0(jyb),rhoh(jyb),jbt,frhot,rhot,frhob,rhob)
      do 380 j=jyb,jyt
  380 rhoh(j)=amax1(rhoh(j),rhomin)
      call sourcd (jbt,dtsub,2,unit(jyb),srvy(jyb),srvyt,srvyb)
      if (lgrav) then
      call sourcd (jbt,dtsub,3,unit(jyb),sgrv(jyb),sgrv(jyt+1),0.0)
```

```
      endif
      call fryfct (rvy0(jyb),rvyh(jyb),jbt,frvyt,rvyt,frvyb,rvyb)
      call fryfct (rvx0(jyb),rvxh(jyb),jbt,frvxt,rvxt,frvxb,rvxb)
      call sourcd (jbt,dtsub,1,unit(jyb),serg(jyb),sergt,sergb)
      call fryfct (erg0(jyb),ergh(jyb),jbt,fergt,ergt,fergb,ergb)
c
c     advance species.
      do 390 j=jyb,jyt
      scr1(j)=1.e-8*rhoh(j)
  390 scr2(j)=0.0
      go to (460,420,400), nspec
  400 call fryfct (rhb0(jyb),rhbh(jyb),jbt,frhbt,rhbt,frhbb,rhbb)
      do 410 j=jyb,jyt
      rhbh(j)=cvmgt(0.0,rhbh(j),rhbh(j).le.scr1(j))
  410 scr2(j)=scr2(j)+rhbh(j)
  420 call fryfct (rha0(jyb),rhah(jyb),jbt,frhat,rhat,frhab,rhab)
      do 430 j=jyb,jyt
      rhah(j)=cvmgt(0.0,rhah(j),rhah(j).le.scr1(j))
  430 scr2(j)=scr2(j)+rhah(j)
      call fryfct (rhc0(jyb),rhch(jyb),jbt,frhct,rhct,frhcb,rhcb)
      do 440 j=jyb,jyt
      rhch(j)=cvmgt(0.0,rhch(j),rhch(j).le.scr1(j))
  440 scr2(j)=scr2(j)+rhch(j)
      do 450 j=jyb,jyt
  450 scrh(j)=rhoh(j)/scr2(j)
  460 continue
c
c     replace variables in 2D arrays.
      do 470 j=jyb,jyt
      rho(i,j)=rhoh(j)
      rvx(i,j)=rvxh(j)
      rvy(i,j)=rvyh(j)
  470 erg(i,j)=ergh(j)
      go to (520,500,480), nspec
  480 do 490 j=jyb,jyt
  490 rhb(i,j)=rhbh(j)*scrh(j)
  500 do 510 j=jyb,jyt
  510 rha(i,j)=rhah(j)*scrh(j)
c
c     loop back for the next col to be integrated.
  520 continue
      return
c
      entry bndrx(bcr,bcl)
c-----------------------------------------------------------------------
c     sets up x boundary conditions for next call to integx.
c       notation :
c          bc = 0   outflow.
c             = 1   subsonic (ambient).
c             =-1   reflective.
c             = 2   prescribed.
c-----------------------------------------------------------------------
c     check if active grid includes boundaries.
      abcl=cvmgt(bcl,0.0,ixl.eq.1)
      abcr=cvmgt(bcr,0.0,ixr.eq.mnx)
c
c     density
      frhor=cvmgt(0.0,1.0,abcr.ge.1.0)
      rhor=cvmgt(rhorbc,0.0,abcr.eq.2.0)
      frhol=cvmgt(0.0,1.0,abcl.ge.1.0)
      rhol=cvmgt(rholbc,0.0,abcl.eq.2.0)
c     species density.
      frhar=1.0
      rhar=0.0
```

```
      frhal=1.0
      rhal=0.0
      frhbr=1.0
      rhbr=0.0
      frhbl=1.0
      rhbl=0.0
      frhcr=frhor
      rhcr=rhor
      frhcl=frhol
      rhcl=rhol
c
c     x - component of momentum.
      frvxr=cvmgt(abcr,amax1(0.0,1.0-abcr),abcr.eq.-1.0)
      rvxr=cvmgt(rvxrbc,0.0,abcr.eq.2.0)
      frvxl=cvmgt(abcl,amax1(0.0,1.0-abcl),abcl.eq.-1.0)
      rvxl=cvmgt(rvxlbc,0.0,abcl.eq.2.0)
c
c     y - component of momentum.
      frvyr=cvmgt(1.0,amax1(0.0,1.0-abcr),abcr.eq.-1.0)
      rvyr=cvmgt(rvyrbc,0.0,abcr.eq.2.0)
      frvyl=cvmgt(1.0,amax1(0.0,1.0-abcl),abcl.eq.-1.0)
      rvyl=cvmgt(rvylbc,0.0,abcl.eq.2.0)
c
c     energy.
      fergr=cvmgt(0.0,1.0,abcr.ge.1.0)
      ergr=cvmgt(ergrbc,0.0,abcr.eq.2.0)
      fergl=cvmgt(0.0,1.0,abcl.ge.1.0)
      ergl=cvmgt(erglbc,0.0,abcl.eq.2.0)
c
      entry bndry(bct,bcb)
c-----------------------------------------------------------------------
c     sets up y boundary conditions for next call to integy.
c-----------------------------------------------------------------------
c     check if active grid includes boundaries.
      abcb=cvmgt(bcb,0.0,jyb.eq.1)
      abct=cvmgt(bct,0.0,jyt.eq.mny)
c
c     density.
      frhot=cvmgt(0.0,1.0,abct.ge.1.0)
      rhot=cvmgt(rh00(jyt+2),0.0,abct.eq.1.0)
      rhot=cvmgt(rhotbc,rhot,abct.eq.2.0)
      frhob=cvmgt(0.0,1.0,abcb.ge.1.0)
      rhob=cvmgt(rh00(jyb),0.0,abcb.eq.1.0)
      rhob=cvmgt(rhobbc,rhob,abcb.eq.2.0)
c
c     species density.
      frhat=1.0
      rhat=0.0
      frhab=1.0
      rhab=0.0
      frhbt=1.0
      rhbt=0.0
      frhbb=1.0
      rhbb=0.0
      frhct=frhot
      rhct=rhot
      frhcb=frhob
      rhcb=rhob
c
c     x - component of momentum.
      frvxt=cvmgt(1.0,amax1(0.0,1.0-abct),abct.eq.-1.0)
      rvxt=cvmgt(rvxtbc,0.0,abct.eq.2.0)
      frvxb=cvmgt(1.0,amax1(0.0,1.0-abcb),abcb.eq.-1.0)
      rvxb=cvmgt(rvxbbc,0.0,abcb.eq.2.0)
```

```
c
c     y - component of momentum.
         frvyt=cvmgt(abct,amax1(0.0,1.0-abct),abct.eq.-1.0)
         rvyt=cvmgt(rvytbc,0.0,abct.eq.2.0)
         frvyb=cvmgt(abcb,amax1(0.0,1.0-abcb),abcb.eq.-1.0)
         rvyb=cvmgt(rvybbc,0.0,abcb.eq.2.0)
c
c     energy.
         fergt=cvmgt(0.0,1.0,abct.ge.1.0)
         ergt=cvmgt(er00(jyt+2),0.0,abct.eq.1.0)
         ergt=cvmgt(ergtbc,ergt,abct.eq.2.0)
         fergb=cvmgt(0.0,1.0,abcb.ge.1.0)
         ergb=cvmgt(er00(jyb),0.0,abcb.eq.1.0)
         ergb=cvmgt(ergbbc,ergb,abcb.eq.2.0)
         return
      end
         subroutine diagno (nstep,time)
c-----------------------------------------------------------
c     performs general diagnostics.
c-----------------------------------------------------------
         parameter (mnx=150, mny=200, mxy=202)
         parameter (mnxx=1, mnyy=1)
         parameter (mxx=1)
         parameter (nop=1, nsta=1)
         integer otape
         integer alpha
         logical lmgrid, lstat, lnuc, therml, lgrav, leos
         real rha(mnx,mny), rhb(mnxx,mnyy), rhc(mnxx,mnyy)
         real tem(mnxx,mnyy), sc1(mnxx,mnyy), rho(mnx,mny)
         real rvx(mnx,mny), rvy(mnx,mny), erg(mnx,mny)
         common nx, ny, nxp, nyp, nspec, idump, idiag, lmgrid
         common lstat, lnuc, therml, lgrav, leos
         common alpha, gamma, gm1, dt, dx, dy, rhomin
         common tem, sc1, rho, rvx, rvy, erg, rha, rhb, rhc
c
         real pre(mnx,mny)
         equivalence (pre,rvx)
         real xcor(mxy), xcoro(mxy), ycor(mxy), ycoro(mxy), unit(mxy)
         common /grids/ xcor, xcoro, ycor, ycoro, unit
c
         real xbnd(5), dxr(4), ybnd(3), dyr(2)
         common /grdcon/ ndx, nsmth, mdy, alt, nahed, xfine, hx, hy,
     1   xbnd, ybnd, dxr, dyr, xleft
c
         real rhom(mxy), pres(mxy), vel(mxy), uh(mxy)
         real ergk(mxy), srvx(mxy), srvy(mxy), serg(mxy)
         real rha0(mxy), rhb0(mxy), rhc0(mxy), vh(mxy)
         real rho0(mxy), rvx0(mxy), rvy0(mxy), erg0(mxy)
         real rhoh(mxy), rvxh(mxy), rvyh(mxy), ergh(mxy)
         real delx(mxy), dely(mxy), scrh(mxy), sgrv(mxy), sgam(mxy)
         real rmin(mxy), rmax(mxy), rvr(mxy)
         common /scrch/ rhom, pres, vel, uh, ergk, srvx, srvy, serg,
     1   rha0, rhb0, rhc0, vh, rho0, rvx0, rvy0, erg0, rhoh, rvxh, rvyh
     2   , ergh, delx, dely, scrh, sgrv, sgam, rmin, rmax, rvr
c
         real rscr(mxy), pscr(mxy), gscr(mxy), scr1(mxy), scr2(mxy)
         equivalence (rscr,vel), (pscr,rvr), (gscr,uh), (scr1,vh)
         equivalence (scr2,scrh)
c
         logical part
         real xp(nop), yp(nop)
         common /part/ part, xp, yp, nopp
c
         real rhs(mxx,nsta), vls(mxx,nsta), qms(mxx,nsta), tme(mxx)
```

```
      real prs(mxx,nsta), xs(nsta), ys(nsta)
      common /stat/ rhs, vls, gms, prs, tme, xs, ys, fracn, eblast,
     1      ixx
c
      real pr00(mxy), rh00(mxy), er00(mxy), gp00(mxy), gravy(mxy)
      parameter (jthm=1)
      real fsky(mxy), rhtl(mnx,jthm)
      common /grav/ pr00, rh00, er00, gp00, gravy, fsky, rhtl
     1      , jtherm
c
      logical lscan, rigt, left, ltop, botm
      common /scan/ lscan, rigt, left, ltop, botm
      common /active/ ixr, jxr, ixl, jxl, iyt, jyt, iyb, jyb, ilr,
     1      jbt, iadd, jadd, factor
c
c    compute the various conservation sums for the system.
      call ngridd (xcor,nxp,alpha)
      call ogridd (nxp)
      call ngridd (xcor,nxp,alpha)
      do 30 j=1,ny
      call consrd (rho(1,j),nx,rho0(j))
      call consrd (rvx(1,j),nx,rvx0(j))
      call consrd (rvy(1,j),nx,rvy0(j))
      call consrd (erg(1,j),nx,erg0(j))
      go to (30,20,10), nspec
   10 call consrd (rhb(1,j),nx,rhb0(j))
   20 call consrd (rha(1,j),nx,rha0(j))
   30 continue
      call ngridd (ycor,nyp,1)
      call ogridd (nyp)
      call ngridd (ycor,nyp,1)
      call consrd (rho0,ny,rhosum)
      call consrd (rvx0,ny,rvxsum)
      call consrd (rvy0,ny,rvysum)
      call consrd (erg0,ny,ergsum)
      go to (60,50,40), nspec
   40 call consrd (rhb0,ny,rhbsum)
   50 call consrd (rha0,ny,rhasum)
   60 continue
      write (6,210) nstep,time,rhasum,rhbsum,rhosum,rvxsum,rvysum
     1      ,ergsum
      return
c
      entry pressg
c-----------------------------------------------------------------
c    calculates pressure in the entire grid and stores it in rvx,
c    which is stored temporarily on disk.
c-----------------------------------------------------------------
      call scrdmp (rvx,1)
      do 90 j=1,mny
      do 70 i=1,mnx
   70 ergk(i)=amax1(0.0,erg(i,j)-0.5*(rvx(i,j)*rvx(i,j)+rvy(i,j)*rvy
     1      (i,j))/rho(i,j)-rho(i,j)*gp00(j+1))
      if (leos) then
c       call eos (mnx,nspec,rho(i,j),ergk,sgam,pre(1,j),rscr,pscr,gscr
c     1       ,scr1,scr2,rha(1,j),rhb(1,j))
      call eosp1 (rho(1,j),ergk,mnx,sgam,pre(1,j))
      else
      do 80 i=1,mnx
   80 pre(i,j)=gm1*ergk(i)
      endif
   90 continue
      return
c
```

```
      entry datain(itape)
      read (itape) ix,iy,nspec,imgrid,istat,inuc,igrav,ieos,part
     1 ,therml
      if (ix.eq.nx.and.iy.eq.ny) go to 100
      write (6,220) itape,ix,iy,nx,ny
      stop
  100 continue
      if (.not.istat) go to 110
      read (itape) ixx
      read (itape) ((rhs(i,kk),i=1,mxx),kk=1,nsta)
      read (itape) ((vis(i,kk),i=1,mxx),kk=1,nsta)
      read (itape) ((gms(i,kk),i=1,mxx),kk=1,nsta)
      read (itape) ((prs(i,kk),i=1,mxx),kk=1,nsta)
      read (itape) (xs(ii),ii=1,nsta)
      read (itape) (ys(ii),ii=1,nsta)
      read (itape) (tme(i),i=1,mxx)
  110 read (itape) (xcor(i),i=1,nxp),(ycor(j),j=1,nyp)
      read (itape) iscan,rigt,left,itop,botm,ixr,jxr,ixl,jxl,iyt,jyt
     1 ,iyb,jyb
      go to (140,130,120), nspec
  120 read (itape) ((rhb(i,j),i=1,nx),j=1,ny)
  130 read (itape) ((rha(i,j),i=1,nx),j=1,ny)
  140 continue
      read (itape) ((rho(i,j),i=1,nx),j=1,ny)
      read (itape) ((pre(i,j),i=1,nx),j=1,ny)
      read (itape) ((rvx(i,j),i=1,nx),j=1,ny)
      read (itape) ((rvy(i,j),i=1,nx),j=1,ny)
      read (itape) ((erg(i,j),i=1,nx),j=1,ny)
      if (inuc) read (itape) radl,fracn,eblast
      read (itape) (pr00(i),i=1,mxy)
      read (itape) (rh00(i),i=1,mxy)
      read (itape) (er00(i),i=1,mxy)
      read (itape) (gp00(i),i=1,mxy)
      read (itape) (gravy(i),i=1,mxy)
      read (itape) (fsky(i),i=1,mxy)
      if (therml) read (itape) jtherm,((rhtl(i,j),i=1,mnx),j=1,jthm)
      read (itape) dt
      if (.not.part) go to 150
      read (itape) nopp
      read (itape) (xp(i),i=1,nopp)
      read (itape) (yp(i),i=1,nopp)
  150 continue
      read (itape) (xcoro(i),i=1,nxp),(ycoro(j),j=1,nyp)
      read (itape) (xbnd(i),i=1,5),(ybnd(j),j=1,3)
      read (itape) (dxr(i),i=1,4),(dyr(j),j=1,2)
      return
c
      entry datout(otape)
      write (otape) nx,ny,nspec,imgrid,istat,inuc,igrav,ieos,part
     1 ,therml
      if (.not.istat) go to 160
      write (otape) ixx
      write (otape) ((rhs(i,kk),i=1,mxx),kk=1,nsta)
      write (otape) ((vis(i,kk),i=1,mxx),kk=1,nsta)
      write (otape) ((gms(i,kk),i=1,mxx),kk=1,nsta)
      write (otape) ((prs(i,kk),i=1,mxx),kk=1,nsta)
      write (otape) (xs(ii),ii=1,nsta)
      write (otape) (ys(ii),ii=1,nsta)
      write (otape) (tme(i),i=1,mxx)
  160 write (otape) (xcor(i),i=1,nxp),(ycor(j),j=1,nyp)
      write (otape) iscan,rigt,left,itop,botm,ixr,jxr,ixl,jxl,iyt
     1 ,jyt,iyb,jyb
      go to (190,180,170), nspec
  170 write (otape) ((rhb(i,j),i=1,nx),j=1,ny)
```

```
 180  write (otape) ((rha(i,j),i=1,nx),j=1,ny)
 190  continue
      write (otape) ((rho(i,j),i=1,nx),j=1,ny)
      write (otape) ((pre(i,j),i=1,nx),j=1,ny)
      call scrdmp (rvx,2)
      write (otape) ((rvx(i,j),i=1,nx),j=1,ny)
      write (otape) ((rvy(i,j),i=1,nx),j=1,ny)
      write (otape) ((erg(i,j),i=1,nx),j=1,ny)
      if (inuc) write (otape) radi,fracn,eblast
      write (otape) (pr00(i),i=1,mxy)
      write (otape) (rh00(i),i=1,mxy)
      write (otape) (er00(i),i=1,mxy)
      write (otape) (gp00(i),i=1,mxy)
      write (otape) (gravy(i),i=1,mxy)
      write (otape) (fsky(i),i=1,mxy)
      if (therml) write (otape) jtherm,((rhtl(i,j),i=1,mnx),j=1,jthm
     1 )
      write (otape) dt
      if (.not.part) go to 200
      write (otape) nopp
      write (otape) (xp(i),i=1,nopp)
      write (otape) (yp(i),i=1,nopp)
 200  continue
      write (otape) (xcoro(i),i=1,nxp),(ycoro(j),j=1,nyp)
      write (otape) (xbnd(i),i=1,5),(ybnd(j),j=1,3)
      write (otape) (dxr(i),i=1,4),(dyr(j),j=1,2)
      return
c
  210 format ('0at step ',i4,'  and time ',1pe12.4,'  the conservation s
     1ums are ...',/,'  sum of rha    sum of rhb    sum of rho   ','  sum
     2 of rvx    sum of rvy    sum of erg  '/6e14.6)
  220 format ('0datain input error on itape =',i3,'   the grid ','size o
     1ff tape is ',i3,' by ',i3,'   but fast2d ','expects ',i3,' by ',i3
     2 )
      end
      subroutine scrdmp (arr,io12)
c-----------------------------------------------------
c     temporarily stores data on disk.
c-----------------------------------------------------
      parameter (mnx=150, mny=200, mxy=202)
      real arr(mnx,mny)
      go to (10,20), io12
   10 open (unit=10,file='store',status='new')
      write (10) ((arr(i,j),i=1,mnx),j=1,mny)
      return
   20 rewind (10)
      read (10) ((arr(i,j),i=1,mnx),j=1,mny)
      close (unit=10,status='delete')
      return
      end
      subroutine fryfct (rhoo,rhon,na,rbc,rhorbc,lbc,rholbc)
c
      real rhoo(na), rhon(na)
cd
cd    * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
cd
cd    fryfct (rhoo, rhon, na, rbc, rhorbc, lbc, rholbc)
cd    originator - fry and book       code 4040, nrl          june 1982
cd    does multicoefficient fct with a switch to choose either
cd    of the form      drho/dt = -div (rho*v) + sources       in either
cd    cartesian, cylindrical, or spherical geometry. the finite-differ-
cd    ence grid can be eulerian, sliding rezone, or lagrangian and can
cd    be arbitrarily spaced. the algorithm used is a low-phase error fct
cd    algorithm, vectorized and optimized for speed. a complete descrip-
```

```
cd    tion appears in nrl memo report 3237, "flux-corrected transport
cd    modules for solving generalized continuity equations".
cd
cd    arguments:  in this routine the right boundary radh(np) is half a
cd    cell beyond the last grid point n at rnc(n) and the left boundary
cd    radh(1) is half a cell before the first grid point at rnc(1).
cd    rhoo   real array(na)    grid point densities at start of step  i
cd    rhon   real array(na)    grid point densities at end of step    o
cd    na     integer           number of cells in the system          i
cd    rbc    real              right boundary condition factor        i
cd    rhorbc real              right guard cell value of rho          i
cd    lbc    real              left boundary condition factor         i
cd    rholbc real              left guard cell value of rho           i
cd
cd    language and limitations:  the subroutine jpbfct is a multiple-
cd    entry fortran routine in single precision (32 bits asc).  the asc
cd    parameter statement is used to set symbolically the internal array
cd    dimensions. underflows are possible when the function being trans
cd    ported has many zeroes. the calculations generally misconserve by
cd    one or two bits per cycle. the relative phase and amplitude errors
cd    (for smooth functions) are typically several percent for charac
cd    teristic lengths of 1 - 2 cells (wavelengths of order 10 cells).
cd    shocks are generally accurate to better than 1 percent. this sub-
cd    routine must be compiled with the y option to force storage and
cd    retention of internal variables. alternatively a common block can
cd    be added to accomplish the same end.
cd
cd    entry points:  jhgfct, dibfct, ogridd, ngridd, velocd, sourcd, and
cd    consrd. the detailed documentation (or the listing below) gives
cd    the explanation and use of the arguments to these other entries.
cd
cd    no auxiliary or library routines are called by jpbfct.
cd
cd    * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
c
          parameter (npt=202)
c
          logical lsourc, lmom, logic(2)
          real rbc, lbc, mask1, mask2
          real mask3
          real scrh(npt), rhot(npt), diff(npt), source(npt)
          real flxh(npt), nulh(npt), mulh(npt), adugth(npt)
          real epsh(npt), fsgn(npt), fabs(npt), lnrhot(npt)
          real terp(npt), term(npt), psgn(npt), lorhot(npt)
          real int(npt), flxm(npt), flxp(npt), deltap(npt)
          real rlo(npt), lh(npt), rlh(npt), adudth(npt)
          real vdtodr(npt), gamma(npt)
c
c     common block /fctcom/ contains grid, geometry and cell information
c     to be communicated between subroutines, hydrodynamic utilities,
c     and the main program.
cd    roh    real array(na+1)  old interface locations set by ogridd
cd    rnh    real array(na+1)  new interface locations set by ngridd
          real roh(npt), rnh(npt), roc(npt), rnc(npt)
          real ah(npt), lo(npt), ln(npt), rln(npt)
          common /fctcom/ roh, rnh, roc, rnc, ah, lo, ln, rln
c
          real e(na), r(na)
          real uh(npa)
          integer ind(nsh)
          integer alpha
          real radh(npa)
          real c(na), d(na)
          real rho(na)
```

```
c
      data fctone, roh, source, lsourc, lmom /1.0,npt*1.,npt*0.,
     1 .false.,.false./
      data pi /3.14159265358979323846264338327950288419716939937511/
      data ftpi /4.18879020478638608437904622405767440795898437 5/
      data c1, c2, c3, c4 /0.166666666666666666666666666666666666666
     1 6666666666666666667,0.33333333333333333333333333333333333333
     2 3333333333333333333,0.13333333333333333333333333333333333333
     3 3333333333333333333,0.03333333333333333333333333333333333333
     4 3333333333333333333/
      equivalence (logic(1),lsourc), (logic(2),lmom)
      equivalence (lnrhot(1),lorhot(1))
      equivalence (flxh(1),scrh(1)), (term(1),source(1))
      equivalence (fabs(1),scrh(1)), (terp(1),source(1))
      equivalence (psgn(1),int(1))
c
      lmom=.false.
      go to 10
c     -----------------------------------------------------------------
c
c
      entry jhgfct(rhoo,rhon,na,rbc,rhorbc,lbc,rholbc,e,r)
c
cd    * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
cd
cd    jhgfct (rhoo, rhon, na, rbc, rhorbc, lbc, rholbc, e, r)
cd    description:  this entry allows the flux corrector in the momentum
cd    equation to be limited in such a way as to prevent new maxima
cd    and minima in the pressure.
cd
cd    aruments:  this entry uses the same arguments as jpbfct with the
cd    addition of arguements e and r
cd    e       real array(na)   grid point total energy density        !
cd    r       real array(na)   grid point mass density                !
cd
cd    * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
c
c     calculate the diffusive and convective fluxes.
      lmom=.true.
   10 n=na
      np=n+1
      do 20 i=2,n
      flxh(i)=0.5*adudth(i)*(rhoo(i)+rhoo(i-1))
   20 diff(i)=(rhoo(i)-rhoo(i-1))
      rhol=rhoo(1)*lbc+rholbc
      rhor=rhoo(n)*rbc+rhorbc
      diff(1)=(rhoo(1)-rhol)
      diff(np)=(rhor-rhoo(n))
      flxh(1)=0.5*adudth(1)*(rhoo(1)+rhol)
      flxh(np)=0.5*adudth(np)*(rhor+rhoo(n))
c
c     calculate lambdao*rhot, the transported mass elements.
      do 30 i=1,n
   30 lorhot(i)=lo(i)*rhoo(i)-(flxh(i+1)-flxh(i))
c
c     add in the source terms as appropriate.
      if (.not.lsourc) go to 50
      do 40 i=1,n
   40 lorhot(i)=lorhot(i)+source(i)
   50 continue
      go to 90
c     -----------------------------------------------------------------
c
c
```

```
      entry dibfct(rhoo,rhon,na,rbc,rhorbc,lbc,rholbc)
c
cd    * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
cd
cd    dibfct (rhoo, rhon, na, rbc, rhorbc, lbc, rholbc)
cd    description: this entry performs nonconservative convection. the
cd    equation      d(rho)/dt = - v . grad(rho) + source    is solved.
cd
cd    arguments:  this routine uses the same arguments as jpbfct.
cd
cd    * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
c
c     calculate the diffusive and convective fluxes.
          lmom=.false.
          n=na
          np=n+1
          do 60 i=2,n
          diff(i)=rhoo(i)-rhoo(i-1)
          flxh(i)=vdtodr(i)*diff(i)
   60     diff(i)=nulh(i)*diff(i)
          rhol=rhoo(1)*lbc+rholbc
          rhor=rhoo(n)*rbc+rhorbc
          diff(1)=nulh(1)*(rhoo(1)-rhol)
          diff(np)=nulh(np)*(rhor-rhoo(n))
          flxh(1)=vdtodr(1)*(rhoo(1)-rhol)
          flxh(np)=vdtodr(np)*(rhor-rhoo(n))
c
c     calculate lambdao*rhot, the transported mass elements.
          do 70 i=1,n
   70     lorhot(i)=ln(i)*(rhoo(i)-0.5*(flxh(i+1)+flxh(i)))
c
c     add in the source terms as appropriate.
          if (.not.lsourc) go to 90
          do 80 i=1,n
   80     lorhot(i)=lorhot(i)+source(i)
c
c     calculate the phoenical antidiffusive fluxes here.
   90     continue
          do 100 i=1,n
  100     lnrhot(i)=lorhot(i)+gamma(i+1)*diff(i+1)-gamma(i)*diff(i)
          do 110 i=1,n
  110     rhot(i)=lorhot(i)*rln(i)
          rhotr=rbc*rhot(n)+rhorbc
          rhotl=lbc*rhot(1)+rholbc
c
          do 120 i=2,n
  120     flxh(i)=mulh(i)*(rhot(i)-rhot(i-1))
          flxh(1)=mulh(1)*(rhot(1)-rhotl)
          flxh(np)=mulh(np)*(rhotr-rhot(n))
c
c     diffuse the solution rhot using old fluxes.
          do 130 i=1,n
  130     lnrhot(i)=lorhot(i)+nulh(i+1)*diff(i+1)-nulh(i)*diff(i)
c
c     calculate the transported/diffused density and grid differences.
          do 140 i=1,n
  140     rhot(i)=lnrhot(i)*rln(i)
          rhotr=rbc*rhot(n)+rhorbc
          rhotl=lbc*rhot(1)+rholbc
          do 150 i=2,n
  150     diff(i)=rhot(i)-rhot(i-1)
          diff(1)=rhot(1)-rhotl
          diff(np)=rhotr-rhot(n)
c
```

```
c     calculate the sign and magnitude of the antidiffusive flux.
          do 160 i=1,np
  160     fsgn(i)=sign(fctone,diff(i))
          do 170 i=1,np
  170     fabs(i)=abs(flxh(i))
c
c     calculate the flux-limiting based on the pressure for the momemtum
          if (.not.lmom) go to 280
          do 180 i=1,n
  180     int(i)=e(i)-0.5*rhot(i)*rhot(i)/r(i)
          do 190 i=2,n
  190     deltap(i)=int(i)-int(i-1)
          deltap(1)=int(1)-lbc*int(1)
          deltap(np)=rbc*int(n)-int(n)
          do 200 i=1,n
          flxp(i+1)=-rhot(i)*(1.-sqrt(amax1(0.,1.-2.*r(i)*deltap(i+1)/
     1    (rhot(i)*rhot(i)))))
          flxm(i)=+rhot(i)*(1.-sqrt(amax1(0.,1.+2.*r(i)*deltap(i)/(rhot
     1    (i)*rhot(i)))))
  200     continue
          do 210 i=1,np
  210     deltap(i)=nulh(i)*deltap(i)
          do 220 i=1,n
  220     int(i)=(int(i)*ln(i)+deltap(i+1)-deltap(i))*rln(i)
          do 230 i=2,n
  230     deltap(i)=rhot(i)*(int(i-1)-int(i))
          deltap(1)=rhot(1)*(lbc*int(1)-int(1))
          deltap(np)=rhot(n)*(int(n)-rbc*int(n))
          do 240 i=1,np
  240     psgn(i)=sign(fctone,deltap(i))
          do 250 i=2,n
  250     terp(i)=fsgn(i)*psgn(i)*ln(i)*amin1(psgn(i)*diff(i+1),psgn(i)
     1    *flxm(i-1),psgn(i)*flxp(i+1))
          terp(1)=fsgn(1)*psgn(1)*ln(1)*amin1(psgn(1)*diff(2),psgn(1)
     1    *flxp(2))
          terp(np)=1.0e+75
          do 260 i=1,np
  260     fabs(i)=amin1(terp(i),fabs(i))
          do 270 i=2,n
  270     term(i)=psgn(i)*fsgn(i)*ln(i-1)*amin1(psgn(i)*diff(i-1),psgn(i
     1    )*flxp(i+1),psgn(i)*flxm(i-1))
          term(np)=psgn(np)*fsgn(np)*ln(n)*amin1(psgn(np)*diff(n),psgn
     1    (np)*flxm(n))
          term(1)=1.0e+75
          go to 320
c
c     calculate the flux-limiting changes on the right and the left.
  280     do 290 i=1,n
  290     terp(i)=fsgn(i)*ln(i)*diff(i+1)
          terp(np)=1.0e+75
          rhotrr=rbc*rhot(n-1)+rhorbc
          terp(np)=fsgn(np)*ln(n)*(rhotrr-rhotr)
          do 300 i=1,np
  300     fabs(i)=amin1(terp(i),fabs(i))
          do 310 i=2,np
  310     term(i)=fsgn(i)*ln(i-1)*diff(i-1)
          term(1)=1.0e+75
c
c     terp and term may also be estimated off the ends of the system.
          rhotll=lbc*rhot(2)+rholbc
          term(1)=fsgn(1)*ln(1)*(rhotl-rhotll)
c
c     correct the fluxes completely now.
  320     do 330 i=1,np
```

```
  330     flxh(i)=fsgn(i)*amax1(0.0,amin1(fabs(i),term(i)))
c
c       calculate the new flux-corrected densities.
          do 340 i=1,n
  340     rhon(i)=rln(i)*(lnrhot(i)-(flxh(i+1)-flxh(i)))
          lsourc=.false.
          do 350 i=1,np
  350     source(i)=0.0
          return
c
c       ----------------------------------------------------------------
c
          entry velocd(uh,npa,dt,ncol)
c
cd      * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
cd
cd      velocd (uh, npa, dt)
cd      description:  this entry calculates all velocity-dependant coeffs.
cd      for multicoefficient fct
cd
cd      arguments:
cd      uh      real array(npa)   flow velocity at cell interfaces       !
cd      npa     integer           number of cell interfaces = n + 1      !
cd      dt      real              stepsize for the time integration      !
cd      ncol    integer that specifies the number of n values
cd
cd      * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
c
c       calculate the interface area x velocity differential x dt.
          np=npa
          n=np-1
          do 360 i=1,np
  360     adudth(i)=ah(i)*dt*uh(i)-adugth(i)
c
c       calculate the half-cell epsilon (v*dt/dx)
          do 370 i=1,np
  370     epsh(i)=adudth(i)*rlh(i)
c
c       next calculate the diffusion and antidiffusion coefficients.
c       variation with epsilon means fourth-order accurate phases.
          if (ncol.eq.n) go to 400
          do 380 i=1,np
          gamma(i)=0.2+0.2*epsh(i)*epsh(i)
          nulh(i)=-c4+c3*epsh(i)**2
  380     mulh(i)=c1-c1*epsh(i)**2
          if (ncol.eq.0) go to 420
          do 390 i=1,ncol
          nulh(i)=c1+c2*epsh(i)*epsh(i)
          mulh(i)=0.25-0.5*nulh(i)
  390     gamma(i)=0.0
          go to 420
  400     continue
          do 410 i=1,np
          nulh(i)=c1+c2*epsh(i)*epsh(i)
          mulh(i)=0.25-0.5*nulh(i)
          gamma(i)=0.0
  410     continue
  420     continue
          rdt=1.0/dt
          do 430 i=1,np
          diff(i)=uh(i)-rdt*(rnh(i)-roh(i))
          nulh(i)=lh(i)*nulh(i)
          gamma(i)=lh(i)*gamma(i)
  430     mulh(i)=lh(i)*mulh(i)
```

{ 6×8

```
c
c     now calculate vdtodr for cnvfct.
          dt2=2.0*dt
          dt4=4.0*dt
          do 440 i=2,n
  440     vdtodr(i)=dt4*diff(i)/(rnh(i+1)-rnh(i-1)+roh(i+1)-roh(i-1))
          vdtodr(1)=dt2*diff(1)/(rnh(2)-rnh(1)+roh(2)-roh(1))
          vdtodr(np)=dt2*diff(np)/(rnh(np)-rnh(n)+roh(np)-roh(n))
          return
c
c     ------------------------------------------------------------------
c
      entry setbd(ind,nsh)
c
cd    ******************************************************************
cd
cd    setbd (ind, nsh)
cd    description:  this entry sets some velocity dependent parameters
cd    to zero at the various material boundries to inhibit diffusion
cd    of material across the boundries.
cd
cd    arguments:
cd    ind     integer array(nsh)   end boundary indices for material    !
cd                                 interfaces.
cd    nsh     integer              number of material interfaces.       !
cd
cd    ******************************************************************
c
c     loop over interfaces.
          do 450 is=1,nsh
          i=ind(is)
          adudth(i)=0.0
          nuih(i)=0.0
          muih(i)=0.0
          gamma(i)=0.0
  450     continue
c
          return
c
c     ------------------------------------------------------------------
c
      entry ngridd(radh,npa,alpha)
c
cd    * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
cd
cd    ngride (radh, npa, alpha)
cd    description:  this entry sets new geometry variables and coeffs.
cd
cd    arguments:
cd    radh   real array(npa)   new cell interface positions            !
cd    npa    integer           number of cell interfaces = n + 1       !
cd    alpha  integer           = 1   for cartesian geometry            !
cd                             = 2   for cylindrical geometry          !
cd                             = 3   for spherical geometry            !
cd                             = 4   for nozzle geometry               !
cd
cd    * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
c
c     calculate the new half-cell positions and grid changes.
          np=npa
          n=np-1
          malpha=iabs(alpha)
          do 460 i=1,np
  460     rnh(i)=radh(i)
```

```
      do 470 i=1,n
470   rnc(i)=0.5*(rnh(i)+rnh(i+1))
c
c     calculate the three coordinate systems.
      go to (480,530,560), malpha
c
c     cartesian coordinates.
480   do 490 i=1,np
490   ah(i)=1.0
      if (alpha.gt.0) go to 510
      do 500 i=1,n
500   ln(i)=rnh(2)-rnh(1)
      go to 600
510   do 520 i=1,n
520   ln(i)=rnh(i+1)-rnh(i)
      go to 600
c
c     cylindrical coordinates.
530   do 540 i=1,np
      diff(i)=rnh(i)*roh(i)
540   ah(i)=pi*(roh(i)+rnh(i))
      do 550 i=1,n
550   ln(i)=pi*(diff(i+1)-diff(i))
      go to 600
c
c     spherical coordinates.
560   do 570 i=1,np
      diff(i)=rnh(i)*rnh(i)*rnh(i)
570   scrh(i)=(roh(i)+rnh(i))*roh(i)
      do 580 i=1,np
580   ah(i)=ftpi*(scrh(i)+rnh(i)**2)
      do 590 i=1,n
590   ln(i)=ftpi*(diff(i+1)-diff(i))
c
c     now the geometric variables which are system independant.
600   do 610 i=2,n
610   lh(i)=amin1(ln(i),ln(i-1))
      lh(1)=amin1(ln(1),ln(2))
      lh(np)=amin1(ln(n),ln(n-1))
      do 620 i=1,n
620   rln(i)=1.0/ln(i)
      if (alpha.gt.0) go to 640
      do 630 i=1,np
630   adugth(i)=ah(i)*(rnh(i)-roh(i))
      go to 660
640   do 650 i=1,np
650   adugth(i)=ah(i)*(rnh(i)-roh(i))
660   do 670 i=2,n
670   rlh(i)=0.5*(rln(i)+rln(i+1))
      rlh(1)=rln(1)
      rlh(np)=rln(n)
      return
c
c     ------------------------------------------------------------
c
      entry sourcd(na,dt,modes,c,d,dr,dl)
c
cd    * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
cd
cd    sourcd (na, dt, modes, c, d, dr, dl)
cd    description:  this entry accumulates different source terms.
cd
cd    arguments.
cd    na      integer           number of cells in the system ( = n)    i
```

```
cd    dt      real                stepsize for the time integration     !
cd    modes   integer             = 1   computes + div (d)               !
cd                                = 2   computes + c*grad (d)            !
cd                                = 3   adds + d to the sources          !
cd                                = 4   + div(d) at interfaces           !
cd                                = 5   + c*grad(d) at interfaces        !
cd                                = 6   dummy for now                    !
cd    c       real array(na)      array of source variables at grid pts  !
cd    d       real array(na)      array of source variables at grid pts  !
cd    dr      real                right boundary value of d (if used)    !
cd    dl      real                left boundary value of d (if used)     !
cd
cd    * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
c
          n=na
          np=n+1
          dth=0.5*dt
          dtq=0.25*dt
          go to (680,710,750,770,800,840), modes
c
c     + div(d) is computed conservatively and added to the sources.
  680     do 690 i=2,n
  690     scrh(i)=dth*ah(i)*(d(i)+d(i-1))
          scrh(1)=dth*ah(1)*(dl+dl)
          scrh(np)=dth*ah(np)*(dr+dr)
          do 700 i=1,n
  700     source(i)=source(i)+(scrh(i+1)-scrh(i))
          lsourc=.true.
          return
c
c     + c*grad(d) is computed efficiently and added to the sources.
  710     do 720 i=2,n
  720     scrh(i)=dtq*(d(i)+d(i-1))
          scrh(1)=dtq*(dl+dl)
          scrh(np)=dtq*(dr+dr)
          do 730 i=1,n
  730     diff(i)=scrh(i+1)-scrh(i)
          do 740 i=1,n
  740     source(i)=source(i)+c(i)*(ah(i+1)+ah(i))*diff(i)
          lsourc=.true.
          return
c
c     + d is added to the sources in an explicit formulation.
  750     do 760 i=1,n
  760     source(i)=source(i)+dt*lo(i)*d(i)
          lsourc=.true.
          return
c
c     + div(d) is computed conservatively from interface data.
  770     do 780 i=1,np
  780     scrh(i)=dt*ah(i)*d(i)
          do 790 i=1,n
  790     source(i)=source(i)+(scrh(i+1)-scrh(i))
          lsourc=.true.
          return
c
c     + c*grad(d) is computed using interface data for d.
  800     do 810 i=1,np
  810     scrh(i)=dth*d(i)
          do 820 i=1,n
  820     diff(i)=scrh(i+1)-scrh(i)
          do 830 i=1,n
  830     source(i)=source(i)+c(i)*diff(i)*(ah(i)+ah(i+1))
          lsourc=.true.
```

```
      return
c
c     a dummy for future source terms.
c         . . . .
c         . . . .
  840     continue
          return
c
c     -----------------------------------------------------------------
c
      entry ogridd(npa)
c
cd    . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
cd
cd    ogridd (npa)
cd    description:  this entry copies old grid and geometry variables
cd    into arrays designated to hold them when new values are defined.
cd
cd    arguments:
cu    npa    integer           number of interfaces in the system     i
cd
cd    . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
c
c     copy the previously new grid values to be used as the old grid.
          n=npa-1
          np=npa
          do 850 i=1,n
          lo(i)=ln(i)
          roc(i)=rnc(i)
  850     rlo(i)=rln(i)
          roh(i)=rlo(i)
          do 860 i=1,np
  860     roh(i)=rnh(i)
          return
c
c     -----------------------------------------------------------------
c
      entry consrd(rho,na,csum)
c
cd    . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
cd
cd    consrd (rho, na, csum)
cd    description:  this entry computes the ostensibly conserved sum.
cd
cd    arguments:
cd    rho    real array(na)    cel values for physical variable (rho) i
cd    na     integer           number of cells in the system          i
cd    csum   real              value of the conservation sum of rho   o
cd
cd    . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
cd
c     compute the ostensibly conserved total mass (beware your b.c.)
          n=na
          csum=0.0
          do 870 i=1,n
  870     csum=csum+ln(i)*rho(i)
          return
      end
          subroutine dtset (istep,time,radl,ensum)
c-----------------------------------------------------------------
c     calculates time step as one-half the courant time step
c     to insure positivity for FCI.
c-----------------------------------------------------------------
          parameter (mnx=150, mny=200, mxy=202)
```

```
      parameter (mnxx=1, mnyy=1)
      integer alpha
      logical lmgrid, lstat, lnuc, therml, lgrav, leos
      real rha(mnx,mny), rhb(mnxx,mnyy), rhc(mnxx,mnyy)
      real tem(mnxx,mnyy), sc1(mnxx,mnyy), rho(mnx,mny)
      real rvx(mnx,mny), rvy(mnx,mny), erg(mnx,mny)
      common nx, ny, nxp, nyp, nspec, idump, idiag, lmgrid
      common lstat, lnuc, therml, lgrav, leos
      common alpha, gamma, gm1, dt, dx, dy, rhomin
      common tem, sc1, rho, rvx, rvy, erg, rha, rhb, rhc
c
      real xcor(mxy), xcoro(mxy), ycor(mxy), ycoro(mxy), unit(mxy)
      common /grids/ xcor, xcoro, ycor, ycoro, unit
c
      real rhom(mxy), pres(mxy), vel(mxy), uh(mxy)
      real ergk(mxy), srvx(mxy), srvy(mxy), serg(mxy)
      real rha0(mxy), rhb0(mxy), rhc0(mxy), vh(mxy)
      real rho0(mxy), rvx0(mxy), rvy0(mxy), erg0(mxy)
      real rhoh(mxy), rvxh(mxy), rvyh(mxy), ergh(mxy)
      real delx(mxy), dely(mxy), scrh(mxy), sgrv(mxy), sgam(mxy)
      real rmin(mxy), rmax(mxy), rvr(mxy)
      common /scrch/ rhom, pres, vel, uh, ergk, srvx, srvy, serg,
     1 rha0, rhb0, rhc0, vh, rho0, rvx0, rvy0, erg0, rhoh, rvxh, rvyh
     2 , ergh, delx, dely, scrh, sgrv, sgam, rmin, rmax, rvr
c
      common /active/ ixr, jxr, ixl, jxl, iyt, jyt, iyb, jyb, ilr,
     1 jbt, iadd, jadd, factor
c
      real pr00(mxy), rh00(mxy), er00(mxy), gp00(mxy), gravy(mxy)
      parameter (jthm=1)
      real fsky(mxy), rhtl(mnx,jthm)
      common /grav/ pr00, rh00, er00, gp00, gravy, fsky, rhtl
     1 , jtherm
c
      real rscr(mxy), pscr(mxy), gscr(mxy), scr1(mxy), scr2(mxy)
      equivalence (rscr,rmin), (pscr,rmax), (gscr,rvr), (scr1,sgrv)
      equivalence (scr2,scrh)
      real gams(mxy), vels(mxy), deltat(mxy)
      equivalence (sgam,gams), (srvx,vels), (deltat,srvy)
c
      common /tstep/ dtmin, dtmax
c
      dtold=dt
      vfluid=0.0
      vsound=0.0
      tmstep=dtmax
      it=0
      jt=0
c
c    get delta x and delta y.
      do 10 i=ixl,ixr
   10 delx(i)=xcor(i+1)-xcor(i)
      do 20 j=jyb,jyt
   20 dely(j)=ycor(j+1)-ycor(j)
c
      do 70 j=jyb,jyt
c
c    maximum fluid velocity and sound speed calculation.
      do 30 i=ixl,ixr
      rhom(i)=1.0/rho(i,j)
      vel(i)=(rvx(i,j)*rvx(i,j)+rvy(i,j)*rvy(i,j))*rhom(i)
      ergk(i)=amax1(0.0,erg(i,j)-0.5*vel(i)-rho(i,j)*gp00(j+1))
   30 vel(i)=sqrt(vel(i)*rhom(i))
      if (leos) then
```

```
c           call eos (ilr,nspec,rho(ixl,j),ergk(ixl),gams(ixl),pres(ixl)
c     1      ,rscr(ixl),pscr(ixl),gscr(ixl),scr1(ixl),scr2(ixl),rhaO(ixl)
c     2      ,rhbO(ixl))
            call eospi (rho(ixl,j),ergk(ixl),ilr,gams(ixl),pres(ixl))
            do 40 i=ixl,ixr
   40       vels(i)=sqrt(gams(i)*pres(i)*rhom(i))
            else
            do 50 i=ixl,ixr
            pres(i)=gmi*ergk(i)
   50       vels(i)=sqrt(gamma*pres(i)*rhom(i))
            endif
            do 60 i=ixl,ixr
   60       deltat(i)=amini(delx(i),dely(j))/(vel(i)+vels(i))
            it=ismin(ilr,deltat(ixl),1)+ixl-1
            if (deltat(it).gt.tmstep) go to 70
            tmstep=deltat(it)
            vfluid=vel(it)
            vsound=vels(it)
            jt=j
   70       continue
c
c     determine the time step
            dt=amaxi(dtmin,amini(1.1*dtold,amini(dtmax,0.5*tmstep)))
            dt=cvmgt(0.1*(float(istep)*dt+float(10-istep)*dtmin),dt,istep
     1      .lt.10)
            time=time+dt
            write (6,80) istep,time,dt,vfluid,vsound,it,jt,radl,ixr,jxr
     1      ,ixl,jxl,iyt,jyt,iyb,jyb
            return
c
   80 format ('s',i5,1x,'t=',1pe10.3,'dt=',1pe10.3,'vfs=',2(1pe10.3),'at
     1 ',2i4,'r',1pe10.3,'ijrltb',8i4)
      end
            subroutine restrt (istep,time,restf,labol)
c-------------------------------------------------------
c     reads data from dump file to restart calculation.
c-------------------------------------------------------
            parameter (mnx=150, mny=200, mxy=202)
            character *8 restf, dumpf, dfile
            character *25 path
            character *40 glab
            character *40 label
            character *40 labol
            logical lsave
            integer otape, ofile
            common /inout/ itape, otape, ifile, ofile, lsave, npath, nlab
c
            open (unit=itape,file=restf,status='old')
            read (itape) nfile,dfile,nstep,time,labol
            write (6,60)
            write (6,50) ifile,nstep,time,nfile,dfile,labol
            if (ifile.ne.nfile) go to 30
            if (istep.ne.nstep+1) go to 40
            call datain (itape)
            close (unit=itape,status='keep')
            write (6,60)
            return
c
            entry dumper(istep,time,dumpf,label,path,glab)
c-------------------------------------------------------
c     this entry dumps data at the specified time step.
c-------------------------------------------------------
            write (dfile,100) ofile
            do 10 i=5,7
```

```
      if (dfile(i:i).eq.' ') dfile(i:i)='0'
   10 continue
      dfile(1:8)=dumpf(1:4)//dfile(5:8)
      open (unit=otape,file=dfile,status='new')
      write (6,60)
      write (6,90) ofile,istep,time,dfile,label
      write (otape) ofile,dfile,istep,time,label
c
      call pressg
      call datout (otape)
      close (unit=otape,status='keep')
      if (.not.isave) go to 20
      write (glab,80) path(1:npath),dfile
      ier=1000
      call mass ('save','\wait=on\',0,glab,ierr)
      write (6,70) dfile,ierr
   20 ofile=ofile+1
      write (6,60)
      return
c
c    error exits follow.
   30 write (6,110) ifile,nfile
      stop
   40 write (6,120) istep,nstep
      stop
c
   50 format ('* * *  restart off file ',i2,' at step no.',i6,' and time
     1 =',1pe12.4,/,'        reading file no.',i3,' which is labeled ',a,
     2 ': ',a,/)
   60 format ('1* * * * * * * * * * * * ',)
   70 format (1x,'ierr for ',a,' =',i4/)
   80 format ('&',a,a,'&')
   90 format ('* * *  dumped onto file ',i2,' at step no.',i6,' and time
     1 =',1pe12.4,//,'        this file is labeled ',a,':',1x,a,//)
  100 format (i8)
  110 format ('Ofile required (',i2,') does not equal file found',' ('
     1 ,i2,')',//)
  120 format (1x,'minstp should be last dump step + 1'/1x,'minstp, last
     1dump step=',2i5/1x,'run aborted'/)
      end
      subroutine eos (np,nspec,rho,ein,gam,pre,rscr,pscr,gscr,scr1
     1    ,scrh,rha,rhb)
c-----------------------------------------------------------------------
c    calculates pressure and gamma for upto three species.
c-----------------------------------------------------------------------
      real rho(np), ein(np), gam(np), pre(np), rscr(np), pscr(np)
      real gscr(np), scr1(np), scrh(np), rha(np), rhb(np)
c
c    initialize.
      do 10 jk=1,np
      rscr(jk)=rho(jk)
      pre(jk)=0.0
      gam(jk)=1.0
   10 scr1(jk)=1.0/rscr(jk)
      go to (60,40,20), nspec
c
c    species 3.
   20 call eosp3 (rho,ein,np,gscr,pscr)
      do 30 jk=1,np
      scrh(jk)=rhb(jk)*scr1(jk)
      scrh(jk)=cvmgt(0.0,scrh(jk),scrh(jk).lt.0.1)
      pre(jk)=pre(jk)+pscr(jk)*scrh(jk)
      gam(jk)=gam(jk)+(gscr(jk)-1.0)*scrh(jk)
   30 rscr(jk)=cvmgt(rscr(jk),rscr(jk)-rhb(jk),scrh(jk).eq.0.0)
```

```
c
c     species 2.
   40     call eosp2 (rho,ein,np,gscr,pscr)
          do 50 jk=1,np
          scrh(jk)=rha(jk)*scr1(jk)
          scrh(jk)=cvmgt(0.0,scrh(jk),scrh(jk).lt.0.1)
          pre(jk)=pre(jk)+pscr(jk)*scrh(jk)
          gam(jk)=gam(jk)+(gscr(jk)-1.0)*scrh(jk)
   50     rscr(jk)=cvmgt(rscr(jk),rscr(jk)-rha(jk),scrh(jk).eq.0.0)
c
c     species 1.
   60     call eosp1 (rho,ein,np,gscr,pscr)
          do 70 jk=1,np
          scr1(jk)=rscr(jk)*scr1(jk)
          pre(jk)=amax1(0.0,pre(jk)+pscr(jk)*scr1(jk))
   70     gam(jk)=gam(jk)+(gscr(jk)-1.0)*scr1(jk)
          return
      end
          subroutine eosp1 (rrr,eee,n,gamma,ppp)
c-----------------------------------------------------------------
c     equation of state routine for air.
c
c     input varible definitions.
c           rrr = mass density
c           eee = internal energy per unit volume
c                 (converted for internal use to energy per unit mass)
c             n = number of entries in arrays rrr & eee
c
c     this routine calculates gamma and ppp as follows:
c         gamma = 1.0 + p/(rho*e)
c         ppp   = pressure(units of eee)
c-----------------------------------------------------------------
          parameter (m=64)
c
      dimension rrr(n), eee(n), gamma(n), ppp(n)
      dimension t11(m), t12(m), t21(m), t22(m), rho(m), e(m), int(m)
      dimension p(m), omp(m), q(m), i(m), j(m), tem(m)
      dimension g(8,105), jcy(m), js(m)
      dimension g1(120), g2(120), g3(120), g4(120), g5(120), g6(120)
      dimension g7(120), gf(840)
c
      equivalence (g1(1),g(1,1)), (g2(1),g(1,16)), (g3(1),g(1,31))
      equivalence (g4(1),g(1,46)), (g5(1),g(1,61)), (g6(1),g(1,76))
      equivalence (g7(1),g(1,91)), (int(1),tem(1)), (gf(1),g(1,1))
c
          data rzerol /774.413/
      data xl16e /2.77258872223977448356890818104147911071777734375/
      data rl16e /0.36067376022224095777346519753336906433105468 75/
c-----------------------------------------------------------------
c     g = gamma - 1.0 is stored for 32 bit word machines in powers of
c     16 across for mass density variation and intermediate values
c     1 - 16 for powers of 16 verticaly which represent the internal
c     energy variation.
c
c     16**(2) .ge. rho .ge. 16**(-6)
c     16**(8) .le. e   .le. 16**(15)
c -----------------------------------------------------------------
      data g1 /8*.4222,8*.4152,8*.4110,8*.4081,8*.4058,8*.4040,8*.4024,8
     1 *.4011,8*.3998,8*.3988,8*.3978,8*.3969,8*.3961,8*.3953,8*.3935/
      data g2 /.3918,.3918,.3918,.3918,.3918,.3918,.3918,.3918,.3723,.37
     1 15,.3707,.3699,.3690,.3680,.3663,.3637,.3555,.3538,.3522,.3502,.3
     2 476,.3430,.3344,.3238,.3370,.3370,.3370,.3364,.3347,.3277,.3099,.
     3 2885,.3257,.3227,.3201,.3134,.3062,.3014,.2884,.2591,.3166,.3110,
     4 .3063,.2946,.2831,.2783,.2677,.2358,.3111,.3006,.2940,.2787,.2635
```

```
     5 ..2588,.2502,.2236,.3075,.2906,.2810,.2665,.2466,.2418,.2350,.213
     6 1,.3043,.2819,.2695,.2554,.2317,.2269,.2216,.2038,.2929,.2740,.25
     7 93,.2455,.2206,.2136,.2097,.1955,.2840,.2672,.2500,.2366,.2166,.2
     8 015,.1988,.1879,.2764,.2611,.2429,.2285,.2125,.1890,.1890,.1811,.
     9 2714,.2555,.2384,.2210,.2079,.1818,.1799,.1747,.2669,.2504,.2343,
     $ .2141,.2037,.1822,.1709,.1689,.2624,.2473,.2304,.2096,.1998,.1828
     $ ..1684,.1639/
      data g3 /.2599,.2446,.2268,.2087,.1961,.1834,.1673,.1601,.2401,.21
     1 91,.1972,.1775,.1592,.1444,.1358,.1203,.2002,.1960,.1749,.1536,.1
     2 376,.1252,.1107,.1044,.1911,.1829,.1633,.1420,.1266,.1101,.1012,.
     3 0933,.1950,.1781,.1566,.1415,.1241,.1118,.1009,.0948,.2001,.1789,
     4 .1594,.1443,.1306,.1189,.1095,.1013,.2040,.1826,.1657,.1494,.1338
     5 ..1177,.1081,.0980,.2034,.1854,.1683,.1497,.1322,.1169,.1051,.094
     6 6,.1969,.1855,.1685,.1487,.1304,.1149,.1024,.0916,.1899,.1837,.16
     7 77,.1475,.1287,.1126,.1002,.0900,.1841,.1817,.1667,.1464,.1272,.1
     8 109,.0983,.0888,.1800,.1800,.1659,.1455,.1262,.1097,.0965,.0878,.
     9 1779,.1787,.1657,.1450,.1254,.1087,.0949,.0868,.1773,.1778,.1656,
     $ .1447,.1250,.1080,.0939,.0859,.1783,.1778,.1658,.1448,.1248,.1076
     $ ..0933,.0851/
      data g4 /.1808,.1781,.1667,.1451,.1248,.1074,.0930,.0843,.2134,.20
     1 40,.1978,.1782,.1565,.1368,.1206,.1074,.2210,.2072,.1957,.1739,.1
     2 516,.1312,.1137,.1000,.2245,.2109,.1989,.1772,.1563,.1390,.1247,.
     3 1133,.2299,.2132,.2017,.1795,.1579,.1384,.1221,.1090,.2350,.2157,
     4 .2023,.1798,.1575,.1370,.1197,.1057,.2397,.2194,.2034,.1796,.1572
     5 ..1372,.1205,.1070,.2452,.2227,.2050,.1805,.1576,.1379,.1236,.111
     6 8,.2510,.2256,.2069,.1814,.1581,.1383,.1231,.1103,.2560,.2282,.20
     7 91,.1822,.1585,.1385,.1226,.1083,.2605,.2312,.2111,.1829,.1588,.1
     8 386,.1222,.1070,.2677,.2358,.2129,.1836,.1592,.1386,.1218,.1071,.
     9 2759,.2403,.2145,.1857,.1598,.1389,.1219,.1078,.2834,.2445,.2160,
     $ .1878,.1603,.1394,.1223,.1084,.2905,.2484,.2175,.1898,.1613,.1399
     $ ..1226,.1090/
      data g5 /.2963,.2531,.2199,.1918,.1625,.1407,.1230,.1096,.4323,.35
     1 82,.3109,.2889,.2803,.2706,.2410,.2224,.4610,.4026,.3624,.3212,.2
     2 926,.2551,.2375,.2015,.4199,.3837,.3401,.2979,.2623,.2318,.2108,.
     3 1854,.3924,.3642,.3194,.2760,.2427,.2157,.1902,.1721,.3794,.3479,
     4 .3025,.2673,.2311,.2019,.1842,.1613,.3674,.3448,.2961,.2593,.2255
     5 ..1994,.1785,.1594,.3573,.3443,.2910,.2517,.2293,.2006,.1843,.167
     6 9,.3661,.3438,.2935,.2597,.2336,.2225,.2143,.2116,.3674,.3435,.30
     7 80,.2728,.2606,.2577,.2573,.2573,.3685,.3453,.3210,.3014,.2942,.2
     8 933,.2932,.2932,.3814,.3612,.3341,.3276,.3257,.3253,.3252,.3252,.
     9 3903,.3752,.3570,.3522,.3513,.3510,.3506,.3496,.4012,.3899,.3782,
     $ .3751,.3743,.3741,.3734,.3713,.4155,.4057,.3956,.3930,.3920,.3913
     $ ..3907,.3890/
      data g6 /.4290,.4205,.4118,.4092,.4077,.4065,.4059,.4047,.5411,.53
     1 85,.5359,.5353,.5351,.5350,.5350,.5350,.5823,.5812,.5801,.5797,.5
     2 796,.5797,.5797,.5797,.6096,.6090,.6085,.6082,.6082,.6083,.6083,.
     3 6083,.6308,.6306,.6305,.6303,.6303,.6305,.6305,.6305,.6481,.6483,
     4 .6485,.6483,.6484,.6486,.6487,.6487,.6627,.6632,.6637,.6636,.6637
     5 ..6640,.6640,.6640,.6754,.6761,.6769,.6768,.6770,.6773,.6773,.677
     6 3,.6866,.6875,.6885,.6884,.6886,.6890,.6890,.6890,.6966,.6977,.69
     7 89,.6989,.6991,.6995,.6995,.6995,.7056,.7070,.7083,.7083,.7085,.7
     8 090,.7090,.7090,.7139,.7154,.7169,.7169,.7172,.7176,.7177,.7177,.
     9 7214,.7231,.7248,.7248,.7251,.7256,.7256,.7256,.7285,.7303,.7321,
     $ .7321,.7325,.7330,.7330,.7330,.7350,.7370,.7390,.7390,.7393,.7398
     $ ..7399,.7399/
      data g7 /.7411,.7432,.7453,.7454,.7457,.7463,.7463,.7463,.8069,.81
     1 03,.8138,.8139,.8145,.8152,.8153,.8153,.8454,.8496,.8538,.8540,.8
     2 547,.8556,.8557,.8557,.8727,.8774,.8822,.8825,.8832,.8842,.8843,.
     3 8843,.8938,.8990,.9042,.9046,.9054,.9064,.9065,.9065,.9111,.9166,
     4 .9222,.9226,.9235,.9246,.9247,.9247,.9258,.9316,.9374,.9379,.9387
     5 ..9399,.9400,.9400,.9384,.9445,.9506,.9511,.9520,.9532,.9533,.953
     6 3,.9496,.9559,.9622,.9627,.9637,.9649,.9650,.9650,.9596,.9661,.97
     7 27,.9731,.9741,.9754,.9755,.9755,.9686,.9753,.9821,.9826,.9836,.9
     8 849,.9850,.9850,.9769,.9837,.9906,.9912,.9922,.9936,.9937,.9937,.
```

```
      9 9845,.9915,.9986,.9991,4*.9999,.9915,.9987,6*.9999,.9981,7*.9999/
c-----------------------------------------------------------------------
c     real air eos, table lookup on gilmore data.  (no temp. model)
c     to avoid costly logrithmic functions the table "g" is stored in a
c     form so that the hexadecimal word structure of a 32 bit machine
c     may be exploited.
c     this logic may be transfered to other machines by recalculating
c     the table "g" appropriate to the word arcitecture of that machine.
c     machine dependent functions and key numbers must also be changed.
c-----------------------------------------------------------------------
      ist=0
      nr=n
   10 nst=min0(nr,m)
c
      do 20 ire=1,nst
      rho(ire)=rzeroi*rrr(ist+ire)
      e(ire)=amax1(3.e8,eee(ist+ire)/rrr(ist+ire))
c
c     calculate mass density variation index "i".
      tem(ire)=alog(rho(ire))*rl16e+500.0
      i(ire)=tem(ire)
      omp(ire)=tem(ire)-i(ire)
      i(ire)=502-i(ire)
      p(ire)=1.0-omp(ire)
      i(ire)=max0(i(ire),1)
c
c     calculate internal energy variation index "j".
      tem(ire)=alog(e(ire))*rl16e
      jcy(ire)=ifix(tem(ire))
      tem(ire)=tem(ire)-jcy(ire)
      tem(ire)=exp(xl16e*tem(ire))
      jcy(ire)=jcy(ire)-7
      js(ire)=tem(ire)
      q(ire)=tem(ire)-js(ire)
      j(ire)=js(ire)+15*jcy(ire)
      j(ire)=min0(j(ire),104)
      j(ire)=i(ire)*8+j(ire)
   20 i(ire)=j(ire)-8
      do 30 ire=1,nst
      t11(ire)=gf(i(ire))
      t21(ire)=gf(i(ire)+1)
      t12(ire)=gf(j(ire))
   30 t22(ire)=gf(j(ire)+1)
c
c     calculate gamma by linear interpolation.
      do 40 ire=1,nst
      t12(ire)=t12(ire)-t11(ire)
      t22(ire)=t22(ire)-t21(ire)
   40 gamma(ist+ire)=1.0+omp(ire)*(t11(ire)+q(ire)*t12(ire))+p(ire)*
     1 (t21(ire)+q(ire)*t22(ire))
c
      nr=nr-nst
      ist=ist+nst
      if (nr.gt.0) go to 10
c
c     calculate presure in units of ees
      do 50 ire=1,n
   50 ppp(ire)=amax1(0.0,eee(ire)*(gamma(ire)-1.0))
c
      return
      end
      subroutine eosp2 (rrr,eee,n,gamma,ppp)
c-----------------------------------------------------------------------
c     equation of state for dust.
```

```
c-----------------------------------------------------------------
      real rrr(n), eee(n), gamma(n), ppp(n)
c
      do 10 i=1,n
      gamma(i)=1.0
   10 ppp(i)=0.0
      return
   end
      subroutine radlos (t,xcor,ycor,radl,fracn)
c-----------------------------------------------------------------
c     subroutines radlos and fbloss perform calculations to account
c     for the radiative energy loss from the fireball to the
c     surroundings.
c
c     the calculations involve the following steps:
c     1. call subroutine fbloss to compute the radiative energy loss and
c     2. subtract this energy proportionately from the entire grid.
c
c     arguments:
c      t    = time,
c      dt   = time step,
c      xcor = x coordinates of grid points,
c      ycor = y coordinates of grid points,
c      rho  = fluid density,
c      rvx  = x momentum,
c      rvy  = y momentum,
c      erg  = energy per unit volume and
c      radl = cumulative radiation energy loss.
c      fracn= fraction of yield deposited in grid.
c-----------------------------------------------------------------
      parameter (mnx=150, mny=200, mxy=202)
      parameter (mnxx=1, mnyy=1)
      integer alpha
      logical lmgrid, lstat, lnuc, therml, lgrav, leos
      real rha(mnx,mny), rhb(mnxx,mnyy), rhc(mnxx,mnyy)
      real tem(mnxx,mnyy), scl(mnxx,mnyy), rho(mnx,mny)
      real rvx(mnx,mny), rvy(mnx,mny), erg(mnx,mny)
      common nx, ny, nxp, nyp, nspec, idump, idiag, lmgrid
      common lstat, lnuc, therml, lgrav, leos
      common alpha, gamma, gm1, dt, dx, dy, rhomin
      common tem, scl, rho, rvx, rvy, erg, rha, rhb, rhc
      real xcor(mxy), ycor(mxy)
c
      real rhom(mxy), pres(mxy), vel(mxy), uh(mxy)
      real ergk(mxy), srvx(mxy), srvy(mxy), serg(mxy)
      real rha0(mxy), rhb0(mxy), rhc0(mxy), vh(mxy)
      real rho0(mxy), rvx0(mxy), rvy0(mxy), erg0(mxy)
      real rhoh(mxy), rvxh(mxy), rvyh(mxy), ergh(mxy)
      real delx(mxy), dely(mxy), scrh(mxy), sgrv(mxy), sgam(mxy)
      real rmin(mxy), rmax(mxy), rvr(mxy)
      common /scrch/ rhom, pres, vel, uh, ergk, srvx, srvy, serg,
     1 rha0, rhb0, rhc0, vh, rho0, rvx0, rvy0, erg0, rhoh, rvxh, rvyh
     2 , ergh, delx, dely, scrh, sgrv, sgam, rmin, rmax, rvr
c
      real pr00(mxy), rh00(mxy), er00(mxy), gp00(mxy), gravy(mxy)
      parameter (jthm=1)
      real fsky(mxy), rhtl(mnx,jthm)
      common /grav/ pr00, rh00, er00, gp00, gravy, fsky, rhtl
     1 , jtherm
      common /active/ ixr, jxr, ixl, jxl, iyt, jyt, iyb, jyb, ilr,
     1 jbt, iadd, jadd, factor
c
c     compute the energy radiated by the fireball in dt seconds.
      call fblos1 (t,dt,f)
```

```
      f=f*fracn
      radl=radl+f
      if (f.eq.0.0) return
c
c    subtract this energy proportionately from the entire grid.
      tsum=0.0
      do 20 j=jyb,jyt
      do 10 i=ixl,ixr
      rhom(i)=1.0/rho(i,j)
      ergk(i)=rhom(i)*amax1(0.0,erg(i,j)-0.5*rhom(i)*(rvx(i,j)*rvx(i
     1 ,j)+rvy(i,j)*rvy(i,j))-rho(i,j)*gp00(j+1))
   10 ergk(i)=ergk(i)*ergk(i)*ergk(i)*ergk(i)
      tsum=tsum+ssum(ixr-ixl+1,ergk,1)
   20 continue
      fact=f/tsum
c
      do 30 j=jyb,jyt
      do 30 i=ixl,ixr
      uh(i)=(xcor(i+1)-xcor(i))*(ycor(j+1)-ycor(j))*3.1415926*(xcor
     1 (i)+xcor(i+1))
      rhom(i)=1.0/rho(i,j)
      ergk(i)=rhom(i)*amax1(0.0,erg(i,j)-0.5*rhom(i)*(rvx(i,j)*rvx(i
     1 ,j)+rvy(i,j)*rvy(i,j))-rho(i,j)*gp00(j+1))
      erg(i,j)=cvmgt(erg(i,j),erg(i,j)-fact*ergk(i)*ergk(i)*ergk(i)
     1 *ergk(i)/uh(i),erg(i,j).lt.1.1*er00(j+1))
   30 continue
c
      return
    end
      subroutine fbloss (t,dt,f,w,sigma)
c-----------------------------------------------------------
c   fireball loss routine scaled to sputter data by al sharp.
c            11july72
c
c    t     time in seconds of last hydro cycle
c    dt    hydro time step, tt is the subroutine time after burst.
c    f     energy %ergs< radiated at time t in dt seconds.
c    w     device yield in kt.
c    sigma ratio of density at burst height to density at sea level
c--------------------------------------------------------
c    this entry point is for initialization
c
      real m
c
      if (w.eq.0.0) return
      rhoa=1.225e-03*sigma
      m=1.31737*rhoa**(-0.0432)
      con=0.17e14*sigma**(-0.6244)
      psem=con*w**(0.53)
      tsem=(1.6828e-04-7.62326e-06*alog(sigma)-3.72132e-06*alog
     1 (sigma)**2-1.51933e-07*alog(sigma)**3)*w**(0.3123)
      bmr=2.0*sigma**0.1008
      bml=sigma**0.1189
      tmax=0.038*w**(0.44)*sigma**(0.36)
      tmin=0.00256*w**(0.39)*sigma**(-0.062)
      tmin=amin1(0.99*tmax,tmin)
      pmax=1.49e13*w**(0.59)*sigma**( 0.45)
      pmin=6.82e11*w**(0.54)*sigma**(-1.02)
      pmin=amin1(0.99*pmax,pmin)
      bsem=psem/pmin-1.0
      bnr=11.2586*rhoa**(0.17827)
      bnl=10.0
      if (rhoa.lt.4.0e-06) bnl=5971.13*rhoa**(0.51428)
      btime=0.91981*tmax*rhoa**(-0.072126)
```

```
      bhite=1.0/(0.8*sigma**(0.3372))-1.0
      tr=tmin/tmax
      c=(alog(tmax)-alog(tmin))*tmax/(tmax-tmin)
      b=m*exp(c)/c
      hintmn=pmin/(tr**(-m)*exp(-b*exp(-c*tr)))
      hintmx=pmax/(exp(-b*exp(-c)))
      return
c
      entry fblost(t,dt,f)
c
c     this entry point is called at the start of each hydro cycle.
c
      if (w.eq.0.0) go to 60
      tt=t+0.5*dt
      if (tt.lt.1.0e-07) go to 50
      tsmax=tt/tmax
      if (tt.lt.tmin) go to 20
      if (tsmax-1.0) 10,30,30
 10   betah=1.35
      alph=3.5
      teta=1.0-alog(1.0/tsmax)/alog(1.0/tr)
      tetab=3.1415926*teta**betah
      hint=hintmx+(hintmn-hintmx)*((1.0+cos(tetab))/2.0)**alph
      go to 40
 20   fac=2.0*bhite/((btime/tt)**bnl+(tt/btime)**bnr)
      bfac=1.0/(fac+1.0)
      facm=1.0+(2.0*bsem/((tsem/tt)**bml+(tt/tsem)**bmr))
c
c   this statement converts power out to energy radiated
c
            f=pmin*bfac*facm*1.0e07*dt
      return
 30   hint=hintmx
 40   fac=2.0*bhite/((btime/tt)*bnl+(tt/btime)**bnr)
      bfac=1.0/(fac+1.0)
      facm=1.0+(2.0*bsem/((tsem/tt)**bml+(tt/tsem)**bmr))
      pwt=hint*(tsmax**(-m)*exp(-b*exp(-c*tsmax)))*bfac*facm
            f=pwt*1.0e07*dt
      return
 50   if (tt.lt.1.0e-08) go to 60
      facm=1.0+(2.0*bsem/((tsem/tt)**bml+(tt/tsem)**bmr))
            f=facm*10.0**((alog10(tt)+8.0)*(alog10(pmin)-1.0)+1.0)
      return
 60         f=0.0
      return
      end
      subroutine staton (k,nstep,time)
c------------------------------------------------------------
c     calculates pressures and dynamic pressures at eulerian stations.
c------------------------------------------------------------
      parameter (mnx=150, mny=200, mxy=202)
      parameter (mnxx=1, mnyy=1)
      parameter (mxx=1, nsta=1)
c
      integer alpha
      logical lmgrid, lstat, lnuc, therml, lgrav, leos
      real rha(mnx,mny), rhb(mnxx,mnyy), rhc(mnxx,mnyy)
      real tem(mnxx,mnyy), scl(mnxx,mnyy), rho(mnx,mny)
      real rvx(mnx,mny), rvy(mnx,mny), erg(mnx,mny)
      common nx, ny, nxp, nyp, nspec, idump, idiag, lmgrid
      common lstat, lnuc, therml, lgrav, leos
      common alpha, gamma, gm1, dt, dx, dy, rhomin
      common tem, scl, rho, rvx, rvy, erg, rha, rhb, rhc
c
```

```
      real xcor(mxy), xcoro(mxy), ycor(mxy), ycoro(mxy), unit(mxy)
      common /grids/ xcor, xcoro, ycor, ycoro, unit
c
      common /active/ ixr, jxr, ixl, jxl, iyt, jyt, iyb, jyb, ilr,
     1     jbt, iadd, jadd, factor
c
      real pr00(mxy), rh00(mxy), er00(mxy), gp00(mxy), gravy(mxy)
      parameter (jthm=1)
      real fsky(mxy), rhtl(mnx,jthm)
      common /grav/ pr00, rh00, er00, gp00, gravy, fsky, rhtl
     1     , jtherm
c
      real rhs(mxx,nsta), vls(mxx,nsta), gms(mxx,nsta), tme(mxx)
      real prs(mxx,nsta), xs(nsta), ys(nsta)
      common /stat/ rhs, vls, gms, prs, tme, xs, ys, fracn, eblast,
     1     ixx
c
      ixx=ixx+1
      tme(ixx)=time
      imin=cvmgt(ixl,1,nstep.ne.1)
      imax=cvmgt(ixr,mnx,nstep.ne.1)
      jmin=cvmgt(jyb,1,nstep.ne.1)
      jmax=cvmgt(jyt,mny,nstep.ne.1)
      do 200 j=1,nsta
      if (xs(j).gt.xcor(imax).or.xs(j).lt.xcor(imin)) go to 200
      if (ys(j).gt.ycor(jmax).or.ys(j).lt.ycor(jmin)) go to 200
      do 190 i=imin,imax
      if (xs(j).le.xcor(i).or.xs(j).gt.xcor(i+1)) go to 190
c-------------------------------------------------------------
c   now inside the column, must find the right row
c   must also interpolate the y direction
c   find the index for the y coordinate of the station
c-------------------------------------------------------------
      do 10 js=jmin,jmax
   10 if (ys(j).ge.ycor(js).and.ys(j).lt.ycor(js+1)) i=js
      yfrac=(ys(j)-ycor(i))/(ycor(i+1)-ycor(i))
      yfrac=yfrac-0.5
      if (yfrac) 20,30,40
   20 m=i
      i=i-1
      if (m.eq.1) i=1
      go to 50
   30 m=i
      go to 50
   40 m=i+1
   50 continue
      fay=(ys(j)-ycor(1))/(ycor(2)-ycor(1))
      if (m.eq.1) go to 60
      ya=(ycor(m+1)+ycor(m))*0.5
      yb=(ycor(m-1)+ycor(m))*0.5
      fay=(ys(j)-yb)/(ya-yb)
c
c   must get the x fraction of the cell
   60 xfrac=(xs(j)-xcor(i))/(xcor(i+1)-xcor(i))
      xfrac=xfrac-0.5
      if (xfrac) 70,110,150
c
c   lower part of the cell
   70 if (i.eq.1) go to 110
      xl=(xcor(i)+xcor(i-1))*0.5
      xr=(xcor(i)+xcor(i+1))*0.5
      fac=(xs(j)-xl)/(xr-xl)
      p1=erg(i-1,1)+(erg(i,1)-erg(i-1,1))*fac
      p2=erg(i-1,m)+(erg(i,m)-erg(i-1,m))*fac
```

```
      ergs=p1+(p2-p1)*fay
      r1=rho(i-1,l)+(rho(i,l)-rho(i-1,l))*fac
      r2=rho(i-1,m)+(rho(i,m)-rho(i-1,m))*fac
      rhos=r1+(r2-r1)*fay
      rhas=0.0
      rhbs=0.0
      rhcs=rhos
      go to (100,90,80), nspec
   80 r1=rhb(i-1,l)+(rhb(i,l)-rhb(i-1,l))*fac
      r2=rhb(i-1,m)+(rhb(i,m)-rhb(i-1,m))*fac
      rhbs=r1+(r2-r1)*fay
      rhcs=rhcs-rhbs
   90 r1=rha(i-1,l)+(rha(i,l)-rha(i-1,l))*fac
      r2=rha(i-1,m)+(rha(i,m)-rha(i-1,m))*fac
      rhas=r1+(r2-r1)*fay
      rhcs=amax1((rhcs-rhas),0.0)
  100 continue
      u1=rvx(i-1,l)+(rvx(i,l)-rvx(i-1,l))*fac
      u2=rvx(i-1,m)+(rvx(i,m)-rvx(i-1,m))*fac
      rvxs=u1+(u2-u1)*fay
      v1=rvy(i-1,l)+(rvy(i,l)-rvy(i-1,l))*fac
      v2=rvy(i-1,m)+(rvy(i,m)-rvy(i-1,m))*fac
      rvys=v1+(v2-v1)*fay
      go to 190
c-----------------------------------------------------------
c     here we are at cell center, so no interpolation is needed
c     in the x direction, but alas must do the y
c-----------------------------------------------------------
  110 p1=erg(i,l)
      p2=erg(i,m)
      ergs=p1+(p2-p1)*fay
      r1=rho(i,l)
      r2=rho(i,m)
      rhos=r1+(r2-r1)*fay
      rhas=0.0
      rhbs=0.0
      rhcs=rhos
      go to (140,130,120), nspec
  120 r1=rhb(i,l)
      r2=rhb(i,m)
      rhbs=r1+(r2-r1)*fay
      rhcs=rhcs-rhbs
  130 r1=rha(i,l)
      r2=rha(i,m)
      rhas=r1+(r2-r1)*fay
      rhcs=amax1((rhcs-rhas),0.0)
  140 continue
      u1=rvx(i,l)
      u2=rvx(i,m)
      rvxs=u1+(u2-u1)*fay
      v1=rvy(i,l)
      v2=rvy(i,m)
      rvys=v1+(v2-v1)*fay
      go to 190
c
c     the upper part of the cell in the x-direction.
  150 xl=(xcor(i)+xcor(i+1))*0.5
      xr=(xcor(i+1)+xcor(i+2))*0.5
      fac=(xs(j)-xl)/(xr-xl)
      p1=erg(i,l)+(erg(i+1,l)-erg(i,l))*fac
      p2=erg(i,m)+(erg(i+1,m)-erg(i,m))*fac
      ergs=p1+(p2-p1)*fay
      r1=rho(i,l)+(rho(i+1,l)-rho(i,l))*fac
      r2=rho(i,m)+(rho(i+1,m)-rho(i,m))*fac
```

```
      rhos=r1+(r2-r1)*fay
      rhas=0.0
      rhbs=0.0
      rhcs=rhos
      go to (180,170,160), nspec
  160 r1=rhb(i,l)+(rhb(i+1,l)-rhb(i,l))*fac
      r2=rhb(i,m)+(rhb(i+1,m)-rhb(i,m))*fac
      rhbs=r1+(r2-r1)*fay
      rhcs=rhcs-rhbs
  170 r1=rha(i,l)+(rha(i+1,l)-rha(i,l))*fac
      r2=rha(i,m)+(rha(i+1,m)-rha(i,m))*fac
      rhas=r1+(r2-r1)*fay
      rhcs=amax1((rhcs-rhas),0.0)
  180 continue
      u1=rvx(i,l)+(rvx(i+1,l)-rvx(i,l))*fac
      u2=rvx(i,m)+(rvx(i+1,m)-rvx(i,m))*fac
      rvxs=u1+(u2-u1)*fay
      v1=rvy(i,l)+(rvy(i+1,l)-rvy(i,l))*fac
      v2=rvy(i,m)+(rvy(i+1,m)-rvy(i,m))*fac
      rvys=v1+(v2-v1)*fay
  190 continue
      rhs(k,j)=rhos
      rhom=1.0/rhos
      vis(k,j)=0.5*(rvxs*rvxs+rvys*rvys)*rhom
      eint=amax1(0.0,ergs-vis(k,j))
      if (leos) then
c      call eos (1,nspec,rhos,eint,gms(k,j),prs(k,j),rscr,pscr,gscr
c    1  ,scr1,scr2,rhas,rhbs)
      call eospl (rhos,eint,1,gms(k,j),prs(k,j))
      else
      prs(k,j)=gm1*eint
      gms(k,j)=gamma
      endif
      vis(k,j)=sqrt(2.0*vis(k,j)*rhom)
  200 continue
      return
   end
      subroutine blufxy (bcl,bcr,bcb,bct)
c-----------------------------------------------------------------------
c  hydro advance for grid containing bluff bodies.
c-----------------------------------------------------------------------
      parameter (mnx=150, mny=200, mxy=202)
      parameter (nnbd=2, nbp1=nnbd+1)
c
      logical lscan, rigt, ltop, left, botm
      common /scan/ lscan, rigt, left, ltop, botm
      common /active/ ixr, jxr, ixl, jxl, iyt, jyt, iyb, jyb, ilr,
     1  jbt, iadd, jadd, factor
c
      real xcor(mxy), xcoro(mxy), ycor(mxy), ycoro(mxy), unit(mxy)
      common /grids/ xcor, xcoro, ycor, ycoro, unit
c
      real rhom(mxy), pres(mxy), vel(mxy), uh(mxy)
      real ergk(mxy), srvx(mxy), srvy(mxy), serg(mxy)
      real rha0(mxy), rhb0(mxy), rhc0(mxy), vh(mxy)
      real rho0(mxy), rvx0(mxy), rvy0(mxy), erg0(mxy)
      real rhoh(mxy), rvxh(mxy), rvyh(mxy), ergh(mxy)
      real delx(mxy), dely(mxy), scrh(mxy), sgrv(mxy), sgam(mxy)
      real rmin(mxy), rmax(mxy), rvr(mxy)
      common /scrch/ rhom, pres, vel, uh, ergk, srvx, srvy, serg,
     1  rha0, rhb0, rhc0, vh, rho0, rvx0, rvy0, erg0, rhoh, rvxh, rvyh
     2  , ergh, delx, dely, scrh, sgrv, sgam, rmin, rmax, rvr
c
      integer nlef(mny,nbp1), nrig(mny,nbp1), imn(nnbd), imx(nnbd)
```

```
      integer nbot(mnx,nbp1), ntop(mnx,nbp1), jmn(nnbd), jmx(nnbd)
      common /bodint/ nlef, nrig, imn, imx, nbot, ntop, jmn, jmx
c
c     when active grid does not include bodies.
      if (ixr.lt.imin-1.or.jyt.lt.jmin-1.or.ixl.gt.imax+1.or.jyb.gt
     1    .jmax+1) then
      call bndrx (bcr,bcl)
      call integx
      call bndry (bct,bcb)
      call integy
      return
      endif
c
c     adjust active grid boundaries if necessary.
      if (iscan) then
      ixl=cvmgt(max0(imin-iadd,1),ixl,imin.le.ixl.and.imax.ge.ixl)
      ixr=cvmgt(min0(imax+iadd,mnx),ixr,imin.le.ixr.and.imax.ge.ixr)
      jyb=cvmgt(max0(jmin-jadd,1),jyb,jmin.le.jyb.and.jmax.ge.jyb)
      jyt=cvmgt(min0(jmax+jadd,mny),jyt,jmin.le.jyt.and.jmax.ge.jyt)
c
c     reset boundary conditions.
      abcl=cvmgt(bcl,0.0,ixl.eq.1)
      abcr=cvmgt(bcr,0.0,ixr.eq.mnx)
      abcb=cvmgt(bcb,0.0,jyb.eq.1)
      abct=cvmgt(bct,0.0,jyt.eq.mny)
      endif
c
c     save grid boundaries.
      isavl=ixl
      isavr=ixr
      jsavb=jyb
      jsavt=jyt
c
c     row integration.
c     ****************
c     region below bodies.
      if (jmin.gt.jsavb) then
      jyt=jmin-1
      jbt=jyt-jyb+1
      call bndrx (abcr,abcl)
      call integx
      endif
c
c     region including the bodies.
      do 10 j=jmin,jmax
      jyb=j
      jyt=j
      jbt=1
      do 10 n=1,nbodp1
      if (nlef(j,n).ne.0) then
      ixl=nlef(j,n)
      ixr=nrig(j,n)
      ilr=ixr-ixl+1
      aabcl=cvmgt(abcl,-1.0,ixl.eq.isavl)
      aabcr=cvmgt(abcr,-1.0,ixr.eq.isavr)
      call bndrx (aabcr,aabcl)
      call integx
      endif
   10 continue
c
c     region above bodies.
      if (jmax.lt.jsavt) then
      ixl=isavl
      ixr=isavr
```

```
      ilr=ixr-ixl+1
      jyb=jmax+1
      jyt=jsavt
      jbt=jyt-jyb+1
      call bndrx (abcr,abcl)
      call integx
      endif
c
c     column integration.
c     *******************
c     region left of bodies.
      if (imin.gt.isavl) then
      ixl=isavl
      ixr=imin-1
      ilr=ixr-ixl+1
      jyb=jsavb
      jyt=jsavt
      jbt=jyt-jyb+1
      call bndry (abct,abcb)
      call integy
      endif
c
c     region including the bodies.
      do 20 i=imin,imax
      ixl=i
      ixr=i
      ilr=1
      do 20 n=1,nbodp1
      if (nbot(i,n).ne.0) then
      jyb=nbot(i,n)
      jyt=ntop(i,n)
      jbt=jyt-jyb+1
      aabcb=cvmgt(abcb,-1.0,jyb.eq.jsavb)
      aabcb=cvmgt(2.0,aabcb,i.lt.imn(1))
      aabct=cvmgt(abct,-1.0,jyt.eq.jsavt)
      call bndry (aabct,aabcb)
      call integy
      endif
   20 continue
c
c     region to right of bodies.
      if (imax.lt.isavr) then
      jyb=jsavb
      jyt=jsavt
      jbt=jyt-jyb+1
      ixl=imax+1
      ixr=isavr
      ilr=ixr-ixl+1
      call bndry (abct,abcb)
      call integy
      endif
c
c     reset active grid boundaries.
      ixr=isavr
      ixl=isavl
      jyb=jsavb
      jyt=jsavt
      ilr=ixr-ixl+1
      jbt=jyt-jyb+1
      return
c
      entry bluf(nbod)
c-----------------------------------------------------------------
c     initialize bluff body parameters.
```

```
c---------------------------------------------------------------
      nbodp1=nbod+1
      imin=1
      imax=mnx
      jmin=1
      jmax=mny
c
c     body boundaries.
      imn(1)=18
      imx(1)=24
      jmn(1)=1
      jmx(1)=105
      imn(2)=18
      imx(2)=52
      jmn(2)=158
      jmx(2)=164
c
c     initialize body integers to zero.
      do 50 n=1,nbodp1
      do 30 j=1,mny
      nlef(j,n)=0
   30 nrig(j,n)=0
      do 40 i=1,mnx
      nbot(i,n)=0
   40 ntop(i,n)=0
   50 continue
c
c     body integers.
c     left, segment 1.
      do 60 j=1,mny
   60 nlef(j,1)=1
c
c     left and right, segment 2.
      do 70 n=1,nbod
      do 70 j=jmn(n),jmx(n)
      nlef(j,2)=imx(n)+1
   70 nrig(j,2)=mnx
c
c     right, segment 1.
      do 80 n=1,nbod
      do 80 j=jmn(n),jmx(n)
   80 nrig(j,1)=imn(n)-1
      do 90 j=jmx(1)+1,jmn(2)-1
   90 nrig(j,1)=mnx
      do 100 j=jmx(2)+1,mny
  100 nrig(j,1)=mnx
c
c     bottom, segment 1.
      do 110 i=1,imn(1)-1
  110 nbot(i,1)=1
      do 120 i=imn(1),imx(1)
  120 nbot(i,1)=jmx(1)+1
      do 130 i=imx(1)+1,mnx
  130 nbot(i,1)=1
c
c     bottom and top, segment 2.
      do 140 i=imn(2),imx(2)
      nbot(i,2)=jmx(2)+1
  140 ntop(i,2)=mny
c
c     top, segment 1.
      do 150 i=1,imn(1)-1
  150 ntop(i,1)=mny
      do 160 i=imn(2),imx(2)
```

```
160     ntop(i,1)=jmn(2)-1
        do 170 i=imx(2)+1,mnx
170     ntop(i,1)=mny
        return
      end
        subroutine dpshok (rho,rvx,rvy,erg)
c-----------------------------------------------------------------------
c     deposits shock in shock tube.
c-----------------------------------------------------------------------
        parameter (mnx=150, mny=200, mxy=202)
        parameter (nnbd=2, nbp1=nnbd+1)
        real rho(mnx,mny), erg(mnx,mny)
        real rvx(mnx,mny), rvy(mnx,mny), mach
c
        integer nlef(mny,nbp1), nrig(mny,nbp1), imn(nnbd), imx(nnbd)
        integer nbot(mnx,nbp1), ntop(mnx,nbp1), jmn(nnbd), jmx(nnbd)
        common /bodint/ nlef, nrig, imn, imx, nbot, ntop, jmn, jmx
        data p1, p2, r1 /1.013e6,3.42e6,1.19e-3/
        logical leos
        common /bbc/ rhobbc, rvxbbc, rvybbc, ergbbc
c
c     set up shock.
        do 10 j=1,jmx(1)
        do 10 i=1,imn(1)-1
        rho(i,j)=rhobbc
        rvx(i,j)=rvxbbc
        rvy(i,j)=rvybbc
10      erg(i,j)=ergbbc
        return
c
        entry jump(gamma,leos,ergamb)
c-----------------------------------------------------------------------
c     calculate conditions behind shock.
c-----------------------------------------------------------------------
c     compute internal energy, mach number and density.
        p2op1=p2/p1
        ergbbc=p2/(gamma-1.0)
        mach=sqrt(((gamma+1.0)*p2op1+(gamma-1.0))/(2.0*gamma))
        rhobbc=r1*(gamma+1.0)*mach*mach/((gamma-1.0)*mach*mach+2.0)
c
c     iterate to get better gamma and energy.
        gam=gamma
        pre=p1
        if (.not.leos) go to 30
        kount=0
20      if (kount.eq.50) then
        write (6,40)
        stop
        endif
        call eosp1 (rhobbc,ergbbc,1,gam,pre)
        kount=kount+1
        ergbbc=p2/(gam-1.0)
        mach=sqrt(((gam+1.0)*p2op1+(gam-1.0))/(2.0*gam))
        rhobbc=r1*(gam+1.0)*mach*mach/((gam-1.0)*mach*mach+2.0)
        if (abs(pre-p2).gt.1.e-6*p2) go to 20
c
c     determine ambient sound speed.
        call eosp1 (r1,ergamb,1,gam,pre)
30      camb=sqrt(gam*pre/r1)
c
c     determine conditions behind shock.
        rvxbbc=0.0
        rvybbc=mach*camb*(rhobbc-r1)
        ergbbc=ergbbc+0.5*rvybbc*rvybbc/rhobbc
```

```
      write (6,50) rhobbc,rvxbbc,rvybbc,ergbbc,mach
      return
c
 40 format (1x,'no convergence to shock conditions')
 50 format (1x,'rho,rvx,rvy,erg,mach=',5e12.5)
    end
      subroutine inital (bcl,bcr,bcb,bct)
      parameter (mnx=150, mny=200, mxy=202)
      parameter (mnxx=1, mnyy=1)
      integer alpha
      logical lmgrid, lstat, lnuc, therml, lgrav, leos
      real rha(mnx,mny), rhb(mnxx,mnyy), rhc(mnxx,mnyy)
      real tem(mnxx,mnyy), sc1(mnxx,mnyy), rho(mnx,mny)
      real rvx(mnx,mny), rvy(mnx,mny), erg(mnx,mny)
      common nx, ny, nxp, nyp, nspec, idump, idiag, lmgrid
      common lstat, lnuc, therml, lgrav, leos
      common alpha, gamma, gm1, dt, dx, dy, rhomin
      common tem, sc1, rho, rvx, rvy, erg, rha, rhb, rhc
c
      real xcor(mxy), xcoro(mxy), ycor(mxy), ycoro(mxy), unit(mxy)
      common /grids/ xcor, xcoro, ycor, ycoro, unit
c
      real rhom(mxy), pres(mxy), vel(mxy), uh(mxy)
      real ergk(mxy), srvx(mxy), srvy(mxy), serg(mxy)
      real rhaO(mxy), rhbO(mxy), rhcO(mxy), vh(mxy)
      real rhoO(mxy), rvxO(mxy), rvyO(mxy), ergO(mxy)
      real rhoh(mxy), rvxh(mxy), rvyh(mxy), ergh(mxy)
      real delx(mxy), dely(mxy), scrh(mxy), sgrv(mxy), sgam(mxy)
      real rmin(mxy), rmax(mxy), rvr(mxy)
      common /scrch/ rhom, pres, vel, uh, ergk, srvx, srvy, serg,
   1  rhaO, rhbO, rhcO, vh, rhoO, rvxO, rvyO, ergO, rhoh, rvxh, rvyh
   2  , ergh, delx, dely, scrh, sgrv, sgam, rmin, rmax, rvr
c
      real prOO(mxy), rhOO(mxy), erOO(mxy), gpOO(mxy), gravy(mxy)
      parameter (jthm=1)
      real fsky(mxy), rhtl(mnx,jthm)
      common /grav/ prOO, rhOO, erOO, gpOO, gravy, fsky, rhtl
   1  , jtherm
      data gO, rO /980.,6.48e8/
c
      real gams(mxy), ptest(mxy), ygp(mxy)
      equivalence (gams,sgam), (ptest,scrh), (ygp,sgrv)
c
c   compute coordinates & set them into common /grids/.
      call xgridO
      call ygridO
      write (6,220)
      write (6,210) (xcor(i),i=1,nxp)
      write (6,230)
      write (6,210) (ycor(j),j=1,nyp)
c
c   set up ambient atmosphere.
c   first get height in KM.
      nypp=nyp+1
      do 10 j=1,mny
 10   ygp(j+1)=0.5*(ycor(j)+ycor(j+1))*1.e-5
      ygp(1)=1.e-5*(ycor(1)-0.5*(ycor(2)-ycor(1)))
      ygp(nypp)=1.e-5*(ycor(nyp)+0.5*(ycor(nyp)-ycor(ny)))
      jlim=cvmgt(nypp,2,lgrav)
c
c   get density and pressure.
      do 20 j=1,jlim
      call cav61 (ygp(j),rhOO(j),tam,amass)
      rhOO(j)=amax1(rhOO(j),rhomin)
```

```
      pr00(j)=rh00(j)*8.3144e7*tam/amass
   20 continue
c
c     get gravitational acceleration and potential.
      gr2=cvmgt(g0*r0*r0,0.0,lgrav)
      do 30 j=1,nypp
   30 ygp(j)=ygp(j)*1.e5
      do 40 j=1,jlim
      gravy(j)=-gr2/(r0+ygp(j))**2
   40 gp00(j)=gr2*(ygp(j)-ygp(1))/((r0+ygp(1))*(r0+ygp(j)))
c
c     use real air EOS to get better gamma and energy.
      do 50 j=1,jlim
   50 er00(j)=pr00(j)/(gamma-1.0)
      if (.not.leos) go to 90
      kount=0
   60 if (kount.eq.50) then
      write (6,260)
      stop
      endif
      call eospt (rh00,er00,jlim,gams,ptest)
      kount=kount+1
      do 70 j=1,jlim
   70 er00(j)=pr00(j)/(gams(j)-1.0)
      do 80 j=1,jlim
      if (abs(pr00(j)-ptest(j)).gt.1.0e-6*pr00(j)) go to 60
   80 continue
   90 do 100 j=1,jlim
  100 er00(j)=er00(j)+rh00(j)*gp00(j)
c
c     set values in remaining cells.
      if (bcb.eq.-1.0.or..not.lgrav) then
      jlim=cvmgt(1,nypp,lgrav)
      do 110 j=1,jlim
      rh00(j)=rh00(2)
      pr00(j)=pr00(2)
      gravy(j)=gravy(2)
  110 er00(j)=er00(2)
      endif
c
c     compute "falling sky" term.
      if (lgrav) then
      fsky(1)=pr00(1)
      fsky(2)=pr00(2)
      do 120 j=2,nypp-1
  120 fsky(j+1)=fsky(j-1)+rh00(j)*gravy(j)*(ygp(j+1)-ygp(j-1))
      else
      do 130 j=1,nypp
  130 fsky(j)=pr00(j)
      endif
c
c     print ambient data.
      write (6,240)
      do 140 j=1,nypp
      jj=nypp-j+1
      write (6,250) jj,ygp(jj),rh00(jj),pr00(jj),er00(jj),gp00(jj)
     1 ,gravy(jj),fsky(jj)
  140 continue
c
c     fill fluid variable arrays
      do 150 j=1,mnv
      do 150 i=1,mnx
      rho(i,j)=rh00(j+1)
      rvx(i,j)=0.0
```

```
          rvy(i,j)=0.0
  150     erg(i,j)=er00(j+1)
          go to (200,180,160), nspec
  160     do 170 j=1,mny
          do 170 i=1,mnx
  170     rhb(i,j)=0.0
  180     do 190 j=1,mny
          do 190 i=1,mnx
  190     rha(i,j)=rho(i,j)
  200     continue
c
          return
c
  210 format (1p10e13.4)
  220 format ('0xcor(i) - the x coordinate grid locations ',/,3x)
  230 format ('0ycor(j) - the y coordinate grid locations ',/,3x)
  240 format (3x,'j',7x,'y',17x,'rho(gm/cc)',3x,'pre(dynes/cm**2)',3x,'e
     1rg(ergs/cc)',3x,'gr pot(erg/gm)',3x,'gr acc(cm/sec**2)',3x,'fsky'/
     2 )
  250 format (1x,i3,7(5x,e12.5))
  260 format (1x,'failure to converge in subroutine inital')
      end
          subroutine burst (xcor,ycor,erg)
c-----------------------------------------------------------------------
c     deposits burst energy in 3 zones.
c-----------------------------------------------------------------------
          parameter (mnx=150, mny=200, mxy=202)
          real erg(mnx,mny), xcor(mxy), ycor(mxy)
c
          parameter (mxx=1, nsta=1)
          real rhs(mxx,nsta), vis(mxx,nsta), gms(mxx,nsta), tme(mxx)
          real prs(mxx,nsta), xs(nsta), ys(nsta)
          common /stat/ rhs, vis, gms, prs, tme, xs, ys, fracn, eblast,
     1    ixx
          data pi /3.1415926/, engin/0.0/
          namelist /blast/ burstn,eblast,nicel,njc1,njc2
c
c     read namelist data.
          read (5,blast)
          write (6,blast)
c
c     determine energy per cell.
          vol=pi*(xcor(nicel+1)**2*(ycor(njc2+1)-ycor(
     x     njc1)))
          ez=eblast*4.2e19*burstn/vol
          fracn=1.0
c
c     distribute burst energy.
          do 10 j=njc1,njc2
          delty=ycor(j+1)-ycor(j)
          do 10 i=1,nicel
           vol=pi*(xcor(i+1)+xcor(i))*(xcor(i+1)-xcor(i))*delty
          engin=engin+ ez*vol
   10        erg(i,j)=erg(i,j)+ez
         write(6,99) engin
 99      format(1x,'energy deposited=',1pe12.5,' ergs'/)
          return
      end
          subroutine cav61 (alt,r,t,m)
c-----------------------------------------------------------
c     calculate mass density, temperature and mean molecular weight
c     based on the average model in the "cira 1961" tables.
c
c     data is stored in the following manner:
```

```
c       0 - 120 km in intervals of 5 km,
c     120 - 300 km in intervals of 10 km and
c     300 - 800 km in intervals of 50km.
c-----------------------------------------------------------------
        real rh(53), tn(53), am(53), m
        data rh /1.23e 03,7.48e-04,4.19e-04,2.02e-04,8.89e-05,4.06e-05
     1  ,1.84e-05,8.69e-06,4.07e-06,2.02e-06,1.02e-06,5.54e-07,3.04e-0
     2  7,1.67e-07,8.84e-08,4.37e 08,1.94e-08,8.12e-09,3.12e-09,1.23e-
     3  09,4.78e-10,2.07e-10,9.49e-11,4.77e-11,2.44e-11,6.95a-12,3.07e
     4  -12,1.69e-12,1.11e-12,8.26e 13,6.59e-13,4.73e 13,3.61e-13,2.77
     5  e-13,2.14e-13,1.67e 13,1.30e-13,1.03e 13,8.11e-14,6.45e 14,5.1
     6  5e-14,4.14e-14,3.34e-14,1.23e-14,5.09e-15,2.33e-15,1.17e-15,6.
     7  15e-16,3.45e-16,2.00e-16,1.19e-16,7.22e-17,4.60e-17/
        oata tn /289.3,256.7,222.4,214.1,217.1,222.5,228.8,236.4,247.9
     1  ,260.6,269.6,268.9,257.8,238.6,217.0,198.1,184.6,178.9,181.1,1
     2  92.5,212.1,237.6,265.0,297.6,343.0,569.4,799.1,1015.,1155.,117
     3  7.,1193.,1211.,1227.,1243.,1259.,1274.,1288.,1302.,1314.,1326.
     4  ,1338.,1348.,1359.,1401.,1436.,1466.,7*1474./
        data am /19*28.97,28.91,28.85,28.78,28.72,28.67,28.60,28.43,28
     1  .25,28.09,27.90,27.70,27.47,27.25,27.00,26.75,26.48,26.18,25.8
     2  7,25.55,25.21,24.85,24.50,24.11,23.74,21.80,20.00,18.55,17.48,
     3  16.31,15.70,15.10,14.62,14.02,13.88/
c
c     calculate appropriate index for table look-up.
        p1=.20*amin1(alt,120.0001)
        p2=.10*amin1(amax1(alt-120.0,0.0),180.0001)
        p3=.02*amax1(alt-300.0,0.0)
        i1=p1
        i2=p2
        i3=p3
        i=i1+i2+i3+1
        i=max0(i,1)
        i=min0(i,52)
c
c     calculate atmospheric quantities by linear interpolation.
        p=(p1-i1)+(p2-i2)+(p3-i3)
        omp=1.0-p
        r=rh(i)*omp+rh(i+1)*p
        t=tn(i)*omp+tn(i+1)*p
        m=am(i)*omp+am(i+1)*p
        return
      end
        subroutine dposit (time)
c-----------------------------------------------------------
c     sets up a nuclear blast at (x0,y0,t0) or a HE
c     blast at (x0,y0,0.0) in a 2D cartesian or cylindrical grid.
c-----------------------------------------------------------------
        parameter (mnx=150, mny=200, mxy=202)
        parameter (mnxx=1, mnyy=1)
        parameter (nop=1)
        parameter (mxx=1, nsta=1)
c
        integer alpha
        logical lmgrid, lstat, lnuc, therml, lgrav, leos
        real rha(mnx,mny), rhb(mnxx,mnyy), rhc(mnxx,mnyy),
        real tem(mnxx,mnyy), sc1(mnxx,mnyy), rho(mnx,mny)
        real rvx(mnx,mny), rvy(mnx,mny), erg(mnx,mny)
        common nx, ny, nxp, nyp, nspec, idump, idiag, lmgrid
        common lstat, lnuc, therml, lgrav, leos
        common alpha, gamma, gm1, dt, dx, dy, rhomin
        common tem, sc1, rho, rvx, rvy, erg, rha, rhb, rhc
c
        real rhom(mxy), pres(mxy), vel(mxy), uh(mxy)
        real ergk(mxy), srvx(mxy), srvy(mxy), serg(mxy)
```

```
      real rhaO(mxy), rhbO(mxy), rhcO(mxy), vh(mxy)
      real rhoO(mxy), rvxO(mxy), rvyO(mxy), ergO(mxy)
      real rhoh(mxy), rvxh(mxy), rvyh(mxy), ergh(mxy)
      real delx(mxy), dely(mxy), scrh(mxy), sgrv(mxy), sgam(mxy)
      real rmin(mxy), rmax(mxy), rvr(mxy)
      common /scrch/ rhom, pres, vel, uh, ergk, srvx, srvy, serg,
     1 rhaO, rhbO, rhcO, vh, rhoO, rvxO, rvyO, ergO, rhoh, rvxh, rvyh
     2 , ergh, delx, dely, scrh, sgrv, sgam, rmin, rmax, rvr
c
      real xcor(mxy), xcoro(mxy), ycor(mxy), ycoro(mxy), unit(mxy)
      common /grids/ xcor, xcoro, ycor, ycoro, unit
c
      real prOO(mxy), rhOO(mxy), erOO(mxy), gpOO(mxy), gravy(mxy)
      parameter (jthm=1)
      real fsky(mxy), rhtl(mnx,jthm)
      common /grav/ prOO, rhOO, erOO, gpOO, gravy, fsky, rhtl
     1 , jtherm
c
      real rhs(mxx,nsta), vls(mxx,nsta), gms(mxx,nsta), tme(mxx)
      real prs(mxx,nsta), xs(nsta), ys(nsta)
      common /stat/ rhs, vls, gms, prs, tme, xs, ys, fracn, eblast,
     1 ixx
c
      parameter (mmr=2200)
      real pbl(mmr), rbl(mmr), vbl(mmr), gbl(mmr), ebl(mmr)
      real rad(mmr), vlc(mmr), ein(mmr), den(mmr), pre(mmr)
      real gma(mmr), pss(mmr)
      equivalence (pbl,pre), (rbl,den), (gbl,rad)
      equivalence (vbl,vlc), (ebl,ein)
c
      real pav(mxy), rav(mxy), vav(mxy), gav(mxy), eav(mxy)
      real gams(mxy)
      integer inmin(mxy), inmax(mxy)
      equivalence (rav,rhoh), (pav,rvxh), (vav,rvyh)
      equivalence (gav,ergh), (gams,sgam), (eav,srvx)
c
      logical part
      real xp(nop), yp(nop)
      common /part/ part, xp, yp, nopp
c
      logical lanl
      namelist /depo/ x0,y0,t0,r0,eblast,lanl
c
      read (5,depo)
      write (6,depo)
c---------------------------------------------------------------
c    add up the ambient energy in the grid  this will enable a
c    comparsion of the final energy, and thus what the energy is in the
c    explosion.
c---------------------------------------------------------------
      call ngridd (xcor,nxp,alpha)
      call ogridd (nxp)
      call ngridd (xcor,nxp,alpha)
      do 10 j=1,mny
      call consrd (erg(1,j),nx,ergO(j))
   10 continue
      call ngridd (ycor,nyp,1)
      call ogridd (nyp)
      call ngridd (ycor,nyp,1)
      call consrd (ergO,ny,ambsum)
      if (.not.inuc) go to 30
      if (abs(eblast-1000.).gt.10.) lanl=.false.
      if (lanl) go to 50
c---------------------------------------------------------------
```

```
c     calculate a scale factor for the radius based on the blast energy
c     measured in kt. then evaluate ktnuc at radial intervals
c     of 1/4 the smallest cell dimension. scale r0 and dr
c     according to blast energy.
c-----------------------------------------------------------------------
         rscale=(1.0/eblast)**(1.0/3.0)
         dr=0.25*amin1(dx,dy)*rscale
         nr=(1.2*rscale*r0/dr)+1
         nrp=nr+1
         radd=0.0
         do 20 i=1,nrp
         call ktnucl (t0,radd,ppp,dens,eee,vvv,gamma)
         pbl(i)=ppp
         rbl(i)=dens
         vbl(i)=vvv
         gbl(i)=gamma
   20    radd=radd+dr
         go to 50
c-----------------------------------------------------------------------
c     deposit HE blast energy in the grid using CJDET at
c     1/4 the smallest cell dimension.
c-----------------------------------------------------------------------
   30    rscale=1.0
         dr=0.25*amin1(dx,dy)
         nr=(1.1*r0/dr)+1
         nrp=nr+1
         x=-0.5*dr
         r0inv=1.0/r0
         do 40 i=1,nrp
         x=x+dr
         xpass=x*r0inv
         call cjdet (xpass,dens,vvv,eee,ppp,gamma)
         pbl(i)=ppp
         ebl(i)=eee
         rbl(i)=dens
         vbl(i)=vvv
   40    gbl(i)=gamma
   50    continue
c
c     calculate cell boundaries relative to (x0, y0). place in delx
c     and dely.
         do 60 i=1,nxp
   60    delx(i)=xcor(i)-x0
         do 70 j=1,nyp
   70    dely(j)=ycor(j)-y0
c
c     compute lengths of position vectors (radii) to cell corners.
         do 80 i=1,nxp
   80    rha0(i)=delx(i)*delx(i)
         do 90 j=1,nyp
   90    rhb0(j)=dely(j)*dely(j)
c
c     now calculate cell centers relative to (x0, y0). place in delx and
c     dely.
         do 100 i=1,mnx
  100    uh(i)=0.5*(delx(i)+delx(i+1))
         do 110 i=1,mnx
  110    delx(i)=uh(i)
         do 120 j=1,mny
  120    vh(j)=0.5*(dely(j)+dely(j+1))
         do 130 j=1,mny
  130    dely(j)=vh(j)
c
c     compute lengths of position vectors of the cell centers
```

```
        do 140 i=1,mnx
  140   scrh(i)=delx(i)*delx(i)
        do 150 j=1,mny
  150   sgrv(j)=dely(j)*dely(j)
        if (.not.lnuc) go to 160
        if (lanl) go to 330
c------------------------------------------------------------------
c     calculate flow field at each point by interpolation onto grid.
c     ktnuc provides pressure, density, and velocity, assuming
c     ambient values of 1.01325e6, 1.225e-3, and 0.0, respectively.
c     we scale the ktnuc values to the ambient values set up in inital.
c------------------------------------------------------------------
        pkt=1.01325e6
        rhokt=1.225e-3
  160   r00=r0*rscale/dr+1
        do 290 j=1,mny
        do 170 i=1,mnx
  170   rvr(i)=scrh(i)+sgrv(j)
c
c     pick up variables from 2D arrays.
        do 180 i=1,mnx
        rvx0(i)=rvx(i,j)
        rvy0(i)=rvy(i,j)
        rho0(i)=rho(i,j)
        rhom(i)=1.0/rho0(i)
  180   erg0(i)=erg(i,j)-rho0(i)*gp00(j+1)
c------------------------------------------------------------------
c     interpolate the blast values onto the grid.
c------------------------------------------------------------------
c     find maximum and minimum radius vectors to corners of each cell.
c     scale according to blast energy.
        do 190 i=1,mnx
        vel(i)=rha0(i)+rhb0(j)
        uh(i)=rha0(i+1)+rhb0(j)
        vh(i)=rha0(i)+rhb0(j+1)
        rmin(i)=amin1(vel(i),uh(i),vh(i))
        rmax(i)=amax1(vel(i),uh(i),vh(i))
        vel(i)=rha0(i+1)+rhb0(j+1)
        rmin(i)=amin1(vel(i),rmin(i))
        rmax(i)=amax1(vel(i),rmax(i))
        rmin(i)=sqrt(rmin(i))*rscale
  190   rmax(i)=sqrt(rmax(i))*rscale
c
c     interpolate onto grid by averaging between rmin and rmax.
        do 200 i=1,mnx
        rmax(i)=rmax(i)/dr+1.0
        rmin(i)=rmin(i)/dr+1.0
        inmax(i)=int(rmax(i))
  200   inmin(i)=int(rmin(i))
c
        do 220 i=1,mnx
        kmin=inmin(i)
        kminp=kmin+1
        kmax=inmax(i)
        kmaxp=kmax+1
        eav(i)=0.0
        pav(i)=0.0
        rav(i)=0.0
        vav(i)=0.0
        gav(i)=0.0
c
        if (rmin(i).gt.r00) go to 220
c
        do 210 k=kminp,kmax
```

```
            eav(i)=eav(i)+ebl(k)
            pav(i)=pav(i)+pbl(k)
            rav(i)=rav(i)+rbl(k)
            vav(i)=vav(i)+vbl(k)
  210       gav(i)=gav(i)+gbl(k)
c
c     interpolate at ends of interval (rmin,rmax).
            flow=1.0-rmin(i)+float(inmin(i))
            fhigh=rmax(i)-float(inmax(i))
            eav(i)=eav(i)+ebl(kmin)*flow+ebl(kmaxp)*fhigh
            pav(i)=pav(i)+pbl(kmin)*flow+pbl(kmaxp)*fhigh
            rav(i)=rav(i)+rbl(kmin)*flow+rbl(kmaxp)*fhigh
            vav(i)=vav(i)+vbl(kmin)*flow+vbl(kmaxp)*fhigh
            gav(i)=gav(i)+gbl(kmin)*flow+gbl(kmaxp)*fhigh
c
c     divide by number of points in interval (rmin,rmax).
            fpoint=rmax(i)-rmin(i)+1.0
            eav(i)=eav(i)/fpoint
            pav(i)=pav(i)/fpoint
            rav(i)=rav(i)/fpoint
            vav(i)=vav(i)/fpoint
            gav(i)=gav(i)/fpoint
  220       continue
            if (.not.lnuc) go to 250
c
c     now calculate new values for fluid variables.
            do 230 i=1,mnx
  230       ergk(i)=amax1(0.0,erg0(i)-0.5*rhom(i)*(rvx0(i)*rvx0(i)+rvy0(i)
     1      *rvy0(i)))
            call eospl (rho0,ergk,nx,gams,pres)
            do 240 i=1,mnx
            if (rmin(i).gt.r00) go to 240
            rho0(i)=(rav(i)/rhokt)*rho0(i)
            rho0(i)=amax1(rho0(i),rhomin)
            uh(i)=sqrt(rvr(i))
            rvr(i)=rho0(i)*vav(i)
            rvx0(i)=rvx0(i)+rvr(i)*delx(i)/uh(i)
            rvy0(i)=rvy0(i)+rvr(i)*dely(j)/uh(i)
            pres(i)=(pav(i)/pkt)*pres(i)
            gams(i)=gav(i)
c
            rhom(i)=1.0/rho0(i)
            ergk(i)=(rvx0(i)*rvx0(i)+rvy0(i)*rvy0(i))*0.5*rhom(i)
            erg0(i)=pres(i)/(gams(i)-1.0)+ergk(i)
  240       continue
            go to 270
  250       do 260 i=1,mnx
            if (rmin(i).gt.r00) go to 260
            rho0(i)=amax1(rav(i),rhomin)
            uh(i)=sqrt(rvr(i))
            rvr(i)=rho0(i)*vav(i)
            rvx0(i)=rvx0(i)+rvr(i)*delx(i)/uh(i)
            rvy0(i)=rvy0(i)+rvr(i)*dely(j)/uh(i)
            erg0(i)=eav(i)
  260       continue
c
c     refill fluid variable arrays.
  270       do 280 i=1,mnx
            rho(i,j)=rho0(i)
            rvx(i,j)=rvx0(i)
            rvy(i,j)=rvy0(i)
  280       erg(i,j)=erg0(i)+rho0(i)*gp00(i+1)
  290       continue
            go to (320,100,300), nspec
```

```
 300      do 310 j=1,mny
          do 310 i=1,mnx
          rha(i,j)=cvmgt(rho(i,j),0.0,rho(i,j).gt.rha(i,j))
 310      continue
 320      continue
c
          if (part) call partin (xcor,ycor,nx,ny,y0)
c
c     set time at start of calculation.
          time=0.0
          if (lnuc) time=t0/rscale
          go to 480
c
c     read lanl data for a 1 megaton blast.
 330      continue
          open (unit=7,file='mt500',status='old')
          read (7,500) time,nrp
          rscale=(1.0/eblast)**(1.0/3.0)
          if (abs(time-t0).gt.0.01*time) stop 2222
          tkt=1.25*time*rscale
          do 340 k=1,nrp
          read (7,510) rad(k),vic(k),ein(k),den(k),pre(k)
c          radd=rad(k)*rscale
c          call ktnucl (tkt,radd,pre(k),den(k),ein(k),vic(k),gma(k))
c          ein(k)=pre(k)/((gma(k)-1.0)*den(k))
 340      continue
          close (unit=7,status='keep')
c
c     recalculate gamma and energy using equation of state for air.
          do 350 k=1,nrp
 350      ein(k)=ein(k)*den(k)
          kount=0
 360      kount=kount+1
          if (kount.gt.40) stop 8888
          call eospl (den,ein,nrp,gma,pss)
          do 370 k=1,nrp
 370      ein(k)=pre(k)/(gma(k)-1.0)
          do 380 k=1,nrp
 380      if (abs(pre(k)-pss(k)).gt.1.e-8*pre(k)) go to 360
c
c     store data in terms of momentum and total energy.
          do 390 k=1,nrp
          ein(k)=ein(k)+0.5*vic(k)*vic(k)*den(k)
 390      vic(k)=den(k)*vic(k)
          do 400 k=2,nrp
 400      pre(k)=1.0/(rad(k)-rad(k-1))
c
          do 470 j=1,mny
          if (abs(dely(j)).gt.rad(nrp)) go to 470
          do 410 i=1,mnx
 410      rvr(i)=sqrt(scrh(i)+sgrv(j))
c
c     pick up variables from 2D arrays.
          do 420 i=1,mnx
          rvx0(i)=rvx(i,j)
          rvy0(i)=rvy(i,j)
          rho0(i)=rho(i,j)
 420      erg0(i)=erg(i,j)-rho0(i)*gp00(j+1)
c---------------------------------------------------------------
c     interpolate the blast values onto the grid.
c---------------------------------------------------------------
          do 450 i=1,mnx
          if (rvr(i).gt.rad(nrp)) go to 450
          kp=1
```

```
      do 430 k=2,nrp
      if (rvr(i).lt.rad(k-1).or.rvr(i).gt.rad(k)) go to 430
      kp=k
      go to 440
  430 continue
  440 fact=pre(kp)*(rvr(i)-rad(kp-1))
      rho0(i)=den(kp-1)+(den(kp)-den(kp-1))*fact
      rvx0(i)=vlc(kp-1)+(vlc(kp)-vlc(kp-1))*fact
      rvy0(i)=rvx0(i)*dely(j)/rvr(i)
      rvx0(i)=rvx0(i)*delx(i)/rvr(i)
      erg0(i)=ein(kp-1)+(ein(kp)-ein(kp-1))*fact
  450 continue
c
c    refill 2D arrays.
      do 460 i=1,mnx
      rho(i,j)=rho0(i)
      rvx(i,j)=rvx0(i)
      rvy(i,j)=rvy0(i)
  460 erg(i,j)=erg0(i)+rho0(i)*gp00(j+1)
  470 continue
c
c    calculate fraction of eblast deposited in the grid.
  480 continue
      call ngridd (xcor,nxp,alpha)
      call ogridd (nxp)
      call ngridd (xcor,nxp,alpha)
      do 490 j=1,mny
      call consrd (erg(1,j),nx,erg0(j))
  490 continue
      call ngridd (ycor,nyp,1)
      call ogridd (nyp)
      call ngridd (ycor,nyp,1)
      call consrd (erg0,ny,dpsum)
      fracn=(dpsum-ambsum)/(eblast*4.2e19)
      write (6,520) ambsum,dpsum,fracn
c
      return
c
  500 format (e12.6,i6)
  510 format (5(1x,e14.8))
  520 format (1x,'energy in ambient mesh=',e12.5,' ergs'/1x,'energy afte
     1r deposition=',e12.5,' ergs'/1x,'fraction of eblast deposited (nuc
     2lear case only)=',e12.5//)
    end
      subroutine cjdet (x,rho,u,e,p,gamma)
c-----------------------------------------------------------------
c     evaluation of the cj detonation flow field (spherical)
c     x = r/rc (radius/charge radius)
c     rho = density (g/cc)
c     u = velocity (cm/sec)
c     e = internal energy (ergs/g)
c     p = pressure (dynes/cm**2)
c-----------------------------------------------------------------------
        common /jwl/ r0, pcj, dcj, e0cj, gcj, a, beos, c, r1, r2, w
        dimension xx(56), rhorhn(56), uun(56), een(56)
      data xx /0.00,0.46,0.47,0.48,0.49,0.50,0.51,0.52,0.53,0.54,0.55
     1 ,0.56,0.57,0.58,0.59,0.60,0.61,0.62,0.63,0.64,0.65,0.66,0.67,0.
     2 68,0.69,0.70,0.71,0.72,0.73,0.74,0.75,0.76,0.77,0.78,0.79,0.80,
     3 0.81,0.82,0.83,0.84,0.85,0.86,0.87,0.88,0.89,0.90,0.91,0.92,0.9
     4 3,0.94,0.95,0.96,0.97,0.98,0.99,1.00/
      data rhorhn /0.58336908,0.58329052,0.58353359,0.58547372,0.5877
     1 1509,0.59020287,0.59290010,0.59578818,0.59884679,0.60206354,0.6
     2 0542828,0.60893458,0.61257368,0.61633992,0.62022847,0.62423760,
     3 0.62836248,0.63260019,0.63695151,0.64141273,0.64598429,0.650666
```

```
     4 24,0.65545791,0.66036117,0.66537666,0.67050570,0.67575240,0.681
     5 11676,0.68660545,0.69221991,0.69796503,0.70384878,0.70987409,0.
     6 71605092,0.72238618,0.72888911,0.73557264,0.74244702,0.74952847
     7 ,0.75683498,0.76438427,0.77220094,0.78031456,0.78875953,0.79757
     8 631,0.80681324,0.81654400,0.82684541,0.83783180,0.84965658,0.86
     9 254716,0.87683612,0.89311200,0.91249108,0.93788856,1.00000000/
       data uun /0.00000000,0.00000000,0.00482083,0.01035851,0.0166617
     1 4,0.02356024,0.03094239,0.03874840,0.04691900,0.05541681,0.0642
     2 1171,0.07328302,0.08260766,0.09216960,0.10195599,0.11195995,0.1
     3 2217087,0.13258135,0.14319155,0.15399441,0.16499040,0.17617987,
     4 0.18756266,0.19914278,0.21092318,0.22290809,0.23510681,0.247521
     5 95,0.26016891,0.27305350,0.28618804,0.29959211,0.31327447,0.327
     6 26002,0.34156698,0.35621879,0.37124780,0.38668105,0.40255913,0.
     7 41892713,0.43583086,0.45333302,0.47150749,0.49044237,0.51024151
     8 ,0.53102863,0.55299151,0.57632965,0.60133791,0.62841576,0.65815
     9 610,0.69143391,0.72980273,0.77623236,0.83853370,1.00000000/
       data een /0.39426151,0.39416990,0.39445356,0.39672121,0.3993476
     1 9,0.40227160,0.40545160,0.40886858,0.41250038,0.41633454,0.4203
     2 6077,0.42457420,0.42896569,0.43353057,0.43826503,0.44316903,0.4
     3 4823870,0.45347190,0.45887235,0.46443692,0.47016850,0.47606906,
     4 0.48213992,0.48838571,0.49480942,0.50141495,0.50821042,0.515197
     5 40,0.52238870,0.52978772,0.53740394,0.54525214,0.55333781,0.561
     6 67936,0.57028919,0.57918406,0.58838654,0.59791517,0.60779816,0.
     7 61806703,0.62875247,0.63989770,0.65155333,0.66377920,0.67664534
     8 ,0.69023508,0.70467544,0.72009885,0.73670238,0.75475061,0.77463
     9 543,0.79692996,0.82264954,0.85371912,0.89516944,1.00000000/
c-----------------------------------------------------------------
c     r0 = ambient charge density (g/cc)
c     pcj = cj pressure (megabars)
c     dcj = cj detonation velocity (cm/microsecond)
c     e0cj = cj energy (mbars-cc/cc)
c     gcj,a,beos,c,r1,r2,w=  jwl eos prameters
c-----------------------------------------------------------------
c     calculate cj jump conditions
        g=gcj
        wn=dcj*1.0e+06
        q=e0cj/r0*1.0e+12
        rhon=r0*(g+1.0)/g
        pn=r0*wn*wn/(g+1.0)
        en=q+0.5*wn*wn/((g+1.0)**2)
        un=wn/(g+1.0)
        if (x.ge.0.0.and.x.le.1.0) go to 10
        rho=1.229e-3
        u=0.
        e=2.542e6
        go to 20
c
c      evaluate self-similar cj detonation flow field
   10   rho=rhon*terpl(x,xx,rhorhn,56,0.0,0.0,3hrho,0)
        u=un*terpl(x,xx,uun,56,0.0,0.0,3hvel,0)
        e=en*terpl(x,xx,een,56,0.0,0.0,6henergy,0)
   20   call eosp2 (rho,e,1,gamma,p)
        return
      end
        function terpl (x,a,b,n,c1,c2,z,nj)
c-----------------------------------------------------------------
c     linear interpolation subprogram
c     input
c      x     abcissa input
c      a     abcissa breakpoints, ascending order
c      b     ordinate breakpoints
c      n     no. a,b data points
c      c1    abcissa data increments (mode 3)
c      c2    abcissa data starting pt (mode 3)
```

```
c        z      data overflow message. blank gives no message.
c        nj     interpolation integer for mode 2 operation
c      output
c        terpl linear interpolation result
c      three modes of operation
c        1   normal.  x,a,b known. nj unknown. do loop search reqd.
c        2   x,a,b,nj known. no search reqd.
c        3   quick interpolation scheme for equally spaced abcissa data
c      calls  none
c-----------------------------------------------------------------------
          dimension a(n), b(n)
          nm1=n-1
          j=c1*x+c2
          j=j+nj
          if (j.gt.nm1) j=nm1
          if (j.gt.0) go to 20
          do 10 i=1,nm1
          j=i
          if (x.le.a(i+1)) go to 20
   10     continue
          print 40, z,x
   20     terpl=a(j+1)-a(j)
          if (terpl.eq.0.) go to 30
          terpl=b(j)+(b(j+1)-b(j))*(x-a(j))/terpl
          return
   30     print 50, z
          terpl=alog(0.)
          stop
c
   40 format (40h range exceeded during interpolation of ,a6,12h data.
     1x = ,e15.7)
   50 format (10x,12hterpl error ,a6,24h data. zero denominator./10x,30h
     1error trace by alog(0.) below.)
      end
          subroutine tlayer (rho,rvx,rvy,erg,rhomin,sum)
c--------------------------------------------------------------
c      thermal layer model using sound speed data.
c--------------------------------------------------------------
          parameter (mnx=150, mny=200, mxy=202)
          parameter (jthm=1)
c
          real xcor(mxy), xcoro(mxy), ycor(mxy), ycoro(mxy), unit(mxy)
          common /grids/ xcor, xcoro, ycor, ycoro, unit
c
          real rhom(mxy), pres(mxy), vel(mxy), uh(mxy)
          real ergk(mxy), srvx(mxy), srvy(mxy), serg(mxy)
          real rha0(mxy), rhb0(mxy), rhc0(mxy), vh(mxy)
          real rho0(mxy), rvx0(mxy), rvy0(mxy), erg0(mxy)
          real rhoh(mxy), rvxh(mxy), rvyh(mxy), ergh(mxy)
          real delx(mxy), dely(mxy), scrh(mxy), sgrv(mxy), sgam(mxy)
          real rmin(mxy), rmax(mxy), rvr(mxy)
          common /scrch/ rhom, pres, vel, uh, ergk, srvx, srvy, serg,
     1    rha0, rhb0, rhc0, vh, rho0, rvx0, rvy0, erg0, rhoh, rvxh, rvyh
     2    , ergh, delx, dely, scrh, sgrv, sgam, rmin, rmax, rvr
c
          real gamo(mxy), gamn(mxy), eint(mxy), cs(mxy)
          real xkt(mxy), ratio(mxy), pscr(mxy)
          equivalence (gamo,delx), (gamn,dely)
          equivalence (eint,vel), (cs,vh), (pscr,ergh)
          equivalence (xkt,rha0), (ratio,rhb0)
c
          real csamb(jthm)
          real rho(mnx,mny), rvx(mnx,mny), rvy(mnx,mny), erg(mnx,mny)
c
```

```
      real pr00(mxy), rh00(mxy), er00(mxy), gp00(mxy), gravy(mxy)
      real fsky(mxy), rhtl(mnx,jthm)
      common /grav/ pr00, rh00, er00, gp00, gravy, fsky, rhtl
     1     , jtherm
c
c     calculate sound speed in ambient atmosphere.
      mnx1=mnx-1
      do 10 j=1,jtherm
      rho1=rho(mnx,j)
      rvx1=rvx(mnx,j)
      rvy1=rvy(mnx,j)
      erg1=erg(mnx,j)
c
      eint1=erg1-0.5*(rvx1*rvx1+rvy1*rvy1)/rho1-rho1*gp00(j+1)
      call eospl (rho1,eint1,1,gam1,pres1)
      csamb(j)=sqrt(gam1*pres1/rho1)
   10 continue
      sum=0.0
c
c     set up thermal layer.
      do 80 j=1,jtherm
      do 20 i=1,mnx
   20 xkt(i)=0.5*(xcor(i)+xcor(i+1))
      call bag (xkt,csamb,j,cs)
      do 30 i=1,mnx
      rho0(i)=rho(i,j)
      rvx0(i)=rvx(i,j)
      rvy0(i)=rvy(i,j)
      erg0(i)=erg(i,j)-rho0(i)*gp00(j+1)
      ergk(i)=0.5*(rvx0(i)*rvx0(i)+rvy0(i)*rvy0(i))/rho0(i)
   30 eint(i)=erg0(i)-ergk(i)
      call eospl (rho0,eint,mnx,gamn,pres)
      icount=0
   40 icount=icount+1
      if (icount.gt.40) stop 5555
      do 50 i=1,mnx
      gamo(i)=gamn(i)
      rho0(i)=pres(i)*gamo(i)/(cs(i)*cs(i))
   50 eint(i)=pres(i)/(gamo(i)-1.0)
      call eospl (rho0,eint,mnx,gamn,pscr)
      do 60 i=1,mnx
      if (abs(gamo(i)-gamn(i)).gt.1 e-8*gamo(i)) go to 40
   60 continue
      do 70 i=1,mnx
      erg0(i)=eint(i)+ergk(i)
      sum=sum+(erg0(i)-erg(i,j))
      rho(i,j)=amax1(rho0(i),rhomin)
      rhtl(i,j)=rho(i,j)
   70 erg(i,j)=erg0(i)+rho0(i)*gp00(j+1)
   80 continue
      return
    end
      subroutine bag (xkt,csamb,j,cs)
c-----------------------------------------------------------------------
c     sound speed vs ground range for MINISCALE.
c-----------------------------------------------------------------------
      parameter (mnx=150, mny=200, mxy=202)
      parameter (jthm=1)
      real xkt(mnx), cs(mnx), csamb(jthm)
      data csthm, xmin, xmax /8.327e4,2073.,6645./
c
      do 10 i=1,mnx
   10 cs(i)=cvmgt(csthm,csamb(j),xkt(i).ge.xmin.and.xkt(i).le.xmax)
      return
```

```
      end
         subroutine griddx (nreg,nxp,xval,scrh)
c-----------------------------------------------------------------
c     calculates x - coordinate values for variable grid size.
c-----------------------------------------------------------------
         parameter (mxy=202)
c
         real xbnd(5), dxr(4), ybnd(5), dyr(2)
         common /grdcon/ ndx, nsmth, mdy, alt, nahed, xfine, hx, hy,
     1   xbnd, ybnd, dxr, dyr, xleft
c
         real ibnd(11), xval(nxp), scrh(mxy)
c
c     set up the region boundary indices.
         nx=nxp-1
         ibnd(1)=1
         nreg1=nreg+1
         do 10 ireg=1,nreg
   10    ibnd(ireg+1)=ibnd(ireg)+(xbnd(ireg+1)-xbnd(ireg))/dxr(ireg)
         ibnd(nreg1)=float(nxp)
c
c     now the values of xval are interpolated in the region tables.
         ireg=1
         do 30 i=1,nxp
         do 20 ireg=1,nreg1
   20    scrh(ireg)=cvmgt(float(ireg),float(nreg1),float(i).le.ibnd
     1   (ireg+1))
         ireg=int(scrh(ismin(nreg1,scrh,1)))
c
c     i is in the region ibnd(ireg) to ibnd(ireg+1).
         xval(i)=xbnd(ireg)+(xbnd(ireg+1)-xbnd(ireg))*((float(i)-ibnd
     1   (ireg))/(ibnd(ireg+1)-ibnd(ireg)+1.0e-5))
   30    continue
c
c     smooth the discontinuities in grid size.
         do 60 ismth=1,nsmth
         do 40 i=2,nx
   40    scrh(i)=0.25*(xval(i+1)+2.0*xval(i)+xval(i-1))
         do 50 i=2,nx
   50    xval(i)=scrh(i)
   60    continue
c
         return
      end
         subroutine xgrid (jtherm)
c-----------------------------------------------------------------
c     calculates interface locations for moving grid problems.
c-----------------------------------------------------------------
         parameter (mnx=150, mny=200, mxy=202)
         parameter (mnxx=1, mnyy=1)
         integer alpha
         logical lmgrid, lstat, lnuc, therml, lgrav, leos
         real rha(mnx,mny), rhb(mnxx,mnyy), rhc(mnxx,mnyy)
         real tem(mnxx,mnyy), sci(mnxx,mnyy), rho(mnx,mny)
         real rvx(mnx,mny), rvy(mnx,mny), erg(mnx,mny)
         common nx, ny, nxp, nyp, nspec, idump, idiag, lmgrid
         common lstat, lnuc, therml, lgrav, leos
         common alpha, gamma, gmi, dt, dx, dy, rhomin
         common tem, sci, rho, rvx, rvy, erg, rha, rhb, rhc
c
         real xcor(mxy), xcoro(mxy), ycor(mxy), ycoro(mxy), unit(mxy)
         common /grids/ xcor, xcoro, ycor, ycoro, unit
c
         real xbnd(5), dxr(4), ybnd(5), dyr(2)
```

```
      common /grdcon/ ndx, nsmth, mdy, alt, nahed, xfine, hx, hy,
     1 xbnd, ybnd, dxr, dyr, xleft
c
      real rhom(mxy), pres(mxy), vel(mxy), uh(mxy)
      real ergk(mxy), srvx(mxy), srvy(mxy), serg(mxy)
      real rha0(mxy), rhb0(mxy), rhc0(mxy), vh(mxy)
      real rho0(mxy), rvx0(mxy), rvy0(mxy), erg0(mxy)
      real rhoh(mxy), rvxh(mxy), rvyh(mxy), ergh(mxy)
      real delx(mxy), dely(mxy), scrh(mxy), sgrv(mxy), sgam(mxy)
      real rmin(mxy), rmax(mxy), rvr(mxy)
      common /scrch/ rhom, pres, vel, uh, ergk, srvx, srvy, serg,
     1 rha0, rhb0, rhc0, vh, rho0, rvx0, rvy0, erg0, rhoh, rvxh, rvyh
     2 , ergh, delx, dely, scrh, sgrv, sgam, rmin, rmax, rvr
c
c  save old x - coordinate values.
      do 10 i=1,nxp
   10 xcoro(i)=xcor(i)
c
c  determine shock-ambient interface.
      ihld=1
      jhld=0
      do 30 j=jtherm+2,mny
      do 20 i=1,mnx
   20 scrh(i)=cvmgt(float(i),0.0,erg(i,j).gt.1.1*erg(mnx,j))
      klim=ismax(mnx,scrh,1)
      ihld=max0(ihld,klim)
      jhld=cvmgt(j,jhld,ihld.eq.klim)
   30 continue
      xhld=xcor(ihld)+20.*dx
      xshk0=xbnd(3)-float(nahed)*dx
      if (xhld.lt.xshk0) return
      xtry=amin1(xshk0+dx,xhld)
      xbnd(3)=amin1(xtry+float(nahed)*dx,xbnd(5))
      xbnd(4)=amin1(xbnd(3)+float(nx-ndx)*hx,xbnd(5))
      xbnd(2)=amax1(xbnd(3)-float(ndx)*dx,xbnd(1))
c
c  compute the new grid as it slides along with the shock.
      call griddx (4,nxp,xcor,scrh)
      if (therml) call rhlyr
      return
c
      entry ygrid
      do 40 j=1,nyp
   40 ycoro(j)=ycor(j)
      return
c
      entry xgrid0
      xcor(1)=0.0
      if (lmgrid) go to 60
      do 50 i=2,nxp
   50 xcor(i)=xcor(i-1)+dx
      go to 70
c
c  standard moving grid for hob problems.  shock edge falls within
c  fine grid.
   60 continue
      nhx=int((xfine-float(ndx)*dx)/hx)+1
      nhx=max0(0,nhx)
      xbnd(1)=xleft
      xbnd(2)=xbnd(1)+float(nhx)*hx
      xbnd(3)=xbnd(2)+float(ndx)*dx
      xbnd(4)=xbnd(1)+float(ndx)*dx+float(nx-ndx)*hx
      xbnd(5)=xbnd(4)
      dxr(1)=hx
```

```
      dxr(2)=dx
      dxr(3)=hx
      dxr(4)=dx
      call griddx (4,nxp,xcor,scrh)
   70 do 80 i=1,nxp
   80 xcoro(i)=xcor(i)
      return
c
      entry ygrid0
      ycor(1)=alt
      if (lmgrid) go to 100
c
c     for a nonmoving grid, we use uniform cells in the y-direction.
      do 90 j=2,nyp
   90 ycor(j)=ycor(j-1)+dy
      go to 130
c-----------------------------------------------------------
c     for a moving grid, the lowest mdy cells are uniform and the rest
c     increase geometrically with cell index. hy is the fractional in-
c     crease from cell to cell.
c--------------------------------------------------------------
  100 continue
      mdyp=min0((mdy+1),nyp)
      do 110 j=2,mdyp
  110 ycor(j)=ycor(1)+float(j-1)*dy
c
      fact=1.0+hy
      mdyp2=min0((mdyp+1),nyp)
      dylrg=dy
      do 120 j=mdyp2,nyp
      dylrg=fact*dylrg
  120 ycor(j)=ycor(j-1)+dylrg
  130 do 140 j=1,nyp
  140 ycoro(j)=ycor(j)
      return
      end
      subroutine dustm(istep,time)
c-----------------------------------------------------------------
c     adds dust so as to model the dust pickup behind the shock front.
c     the SOUR model used is based upon Mirel's model. the dust mass
c     added at each step is mdot*dt*area*u*rho(air).
c-----------------------------------------------------------------
      parameter (mnx=150, mny=200, mxy=202)
      parameter (mnxx=1, mnyy=1)
c
      integer alpha
      logical lmgrid, lstat, lnuc, therml, lgrav, leos
      real rha(mnx,mny), rhb(mnxx,mnyy), rhc(mnxx,mnyy)
      real tem(mnxx,mnyy), sci(mnxx,mnyy), rho(mnx,mny)
      real rvx(mnx,mny), rvy(mnx,mny), erg(mnx,mny)
      common nx, ny, nxp, nyp, nspec, idump, idiag, lmgrid
      common lstat, lnuc, therml, lgrav, leos
      common alpha, gamma, gm1, dt, dx, dy, rhomin
      common tem, sci, rho, rvx, rvy, erg, rha, rhb, rhc
c
      real xcor(mxy), xcoro(mxy), ycor(mxy), ycoro(mxy), unit(mxy)
      common /grids/ xcor, xcoro, ycor, ycoro, unit
c
      real rhom(mxy), pres(mxy), vel(mxy), uh(mxy)
      real ergk(mxy), srvx(mxy), srvy(mxy), serg(mxy)
      real rha0(mxy), rhb0(mxy), rhc0(mxy), vh(mxy)
      real rho0(mxy), rvx0(mxy), rvy0(mxy), erg0(mxy)
      real dnew(mxy)
      real rhoh(mxy), rvxh(mxy), rvyh(mxy), ergh(mxy)
```

```
      real delx(mxy), dely(mxy), scrh(mxy), sgrv(mxy), sgam(mxy)
      real rmin(mxy), rmax(mxy), rvr(mxy)
      common /scrch/ rhom, pres, vel, uh, ergk, srvx, srvy, serg,
     1 rhaO, rhbO, rhcO, vh, rhoO, rvxO, rvyO, ergO, rhoh, rvxh, rvyh
     2 , ergh, delx, dely, scrh, sgrv, sgam, rmin, rmax, rvr
c
      real prOO(mxy), rhOO(mxy), erOO(mxy), gpOO(mxy), gravy(mxy)
      parameter (jthm=1)
      real fsky(mxy), rhtl(mnx,jthm)
      common /grav/ prOO, rhOO, erOO, gpOO, gravy, fsky, rhtl
     1 , jtherm
c
      real dirt(mxy), rair(mxy), rhae(mxy), eint(mxy), qnew(mxy)
      real delta(mxy),theta(mxy),delta2(mxy),deltas(mxy)
      real mdot(mxy),tauw(mxy), visrat(mxy), phi(mxy), mach(mxy)
      equivalence (dirt,rhbO), (rair,rhcO), (rhae,rhoO)
      equivalence (eint,srvy) , (phi,rmin), (mach,rmax)
      data con3, ff, edpO /3.509e-7,0.1,50./
      data beta/0.0064/, sigp/2.5/, G/980./, dia/0.01/
      data amua/620./ ,tw/300./, ff/0.01/
c
c   determine location of shock front. return if shock front has
c   propagated less than 8 zones to the right.
c        do 10 i=1,mnx
c   10   scrh(i)=cvmgt(float(i),0.0,rho(i,1).gt.1.2*rho(mnx,1))
c        ishk=ismax(mnx,scrh,1)
c        sum=ssum(mnx,scrh,1)
c        if (sum.lt.1.0) return
c        inject=max0(1,ishk-8)
        inject=mnx
        if (inject.eq.1) return
c
c    mirels model for blowing dust layer
c    solve the ode for delta the boundary layer thickness
c
c    next we need to compute the new mass added per unit time
c    mdot = cft/2xB  where B defined as ln(1+B)=Bxcft/cfO
c     and cft = 2beta x sigp x G x diameter/rhoair/uair**2
c
c        Luck has it that beta, sigp, g and diameter are constants
c           beta = 0.0064
c           sigp = particle density = 2.5 g/cm**3
c           G = gravity = 980 cm/sec**2
c           diameter = particle dia = 0.01 cm
c                 well almost constant...
c
c        cfO = 2tauw/rhoa/uair**2
c     next tauw is defined as
c         tauw = 0.0225 [ mua/rhoa/uair/delta]**0.25 x phi x rhoa x uair**2
c     now you ask where does phi come from??
c     Finally our final definition
c        phi = [rhom/rhoa]**0.75 x [mum/mua]**0.25
c
c
      do 20 i=1,mnx
      rhom(i)=1.0/rho(i,1)
      rhoh(i)=rho(i,1)
      rair(i)=rho(i,1)-rha(i,1)
      vel(i) = rvx(i,1)*rhom(i)
      qnew(i)=(deltas(i)-delta2(i))*rair(i)*vel(i)
      ergk(i)=0.5*(rvx(i,1)*rvx(i,1)+rvy(i,1)*rvy(i,1))*rhom(i)
   20 eint(i)=erg(i,1)-ergk(i)-rho(i,1)*gpOO(2)
      call eospl (rhoh,eint,inject,sgam,pres)
c
```

```
c      obtain the ratio of viscosities, mum/mua
c
        do 25 i=1,mnx
          pres(i)=amax1(pres(i),1.e5)
        mach(i)=vel(i)/sqrt(sgam(i)*pres(i)/rair(i))
         visrat(i)=0.524*abs(mach(i))
 25     continue
c
c      begin the painful process of integration
c
          call ngridd(xcoro,nx+1,2)
          call ogridd(nx+1)
          call ngridd(xcor,nx+1,2)
c
c     obtain the velocities at cell interfaces
c
           do 26 i=2,nx
 26        uh(i)=(vel(i)+vel(i-1))*0.5
           uh(1)=0.0
           uh(nx+1)=uh(nx)
          call velocd(uh,nx+1,dt,nx+1)
c
c       next do the source terms
c
          do 1 i=1,mnx
 1        rha0(i)=-qnew(i)
          call sourcd(mnx,dt,5,rha0,vel,0.0,0.0)
          call sourcd(mnx,dt,5,delta2,pres,pres(mnx),pres(1))
          do 2 i=1,mnx
 2        rha0(i)=-theta(i)*vel(i)**2*rair(i)
          call sourcd(mnx,dt,1,rha0,rha0,0.0,0.0)
          do 3 i=1,mnx
 3        rha0(i)=qnew(i)*vel(i)
          call sourcd(mnx,dt,1,rha0,rha0,0.0,0.0)
          do 4 i=1,mnx
 4        rha0(i)=tauw(i)
          call sourcd(mnx,dt,3,rha0,rha0,0.0,0.0)
c
          call fryfct(qnew,rhae,mnx,1.0,0.0,1.0,0.0)
c
c         update the qnew array and compute the boundary layer
c         thicknesses,  deltas and delta.
c
        delty=ycor(2)-ycor(1)
          do 5 i=1,mnx
          qnew(i)=rhae(i)
         if(vel(i).eq.0.0) vel(i)=1.e-30
          deltas(i)=qnew(i)/(rair(i)*vel(i)) + delta2(i)
 5        delta(i)=amax1(.02*delty,8.*deltas(i))
c
          do 6 i=1,mnx
          mdot(i)=0.0
          if(vel(i).lt.1.e-4) go to 6
          ta=con3*pres(i)/rair(i)
          rhor=0.5*(tw/ta+1.0) + 0.11*(sgam(i)-1.0)*mach(i)**2
          phi(i)=(1.0/rhor)**0.75*(visrat(i))**0.25
          if(delta(i).lt.1.e-4)  then
          tauw(i)=0.0
          else
          tauw(i)=0.0225*(amua/(rair(i)*vel(i)*delta(i))**0.25)*
     x    phi(i)*rair(i)*vel(i)**2
           endif
          cfo=2.*tauw(i)/rair(i)/vel(i)**2
          cft=2.*beta*sign*G*dia/(rair(i)*vel(i)**2)
```

```
c     solve for B , ln(1+B)=B*cft/cfo
           raat=cfo/cft
             if(raat.le.1.0) go to 6
         B=raat*15.
         do 7 k=1,25
         a=B/(alog(1.0+B))
         B=raat-a+B
 7       continue
          mdot(i)=abs(cft*0.5*B*(rair(i)*vel(i)))
 6        continue
c
c     deposit dust in the flow field behind shock front.
          do 30 i=1,inject
          if(delta(i).lt.1.e-4) then
          dirt(i)=0.0
          else
          dirt(i)=mdot(i)*ff*dt/delty
          endif
          dnew(i)=rha(i,1)+dirt(i)
          rha(i,1)=cvmgt(dnew(i),rha(i,1),ergk(i).gt.edp0)
          rho(i,1)=rair(i)+rha(i,1)
          erg0(i)=amax1(0.0,eint(i)+1.e7*dirt(i)*con3*pres(i)/rho(i,1))
          erg(i,1)=cvmgt(erg0(i)+rho(i,1)*gp00(2)+ergk(i)/(rhom(i)*rho(i
     1    ,1)),erg(i,1),ergk(i).gt.edp0)
   30     continue
          if(mod(istep-1,100).eq.0) then
          write(6,57) istep, time
 57       format(1x,'dirt for step',i5,' and time ',1pe12.5)
            write(6,56) (dirt(i),i=11,30)
            else
            endif
 56        format(1x,1p8e12.5)
          return
c
          entry dust01
c
c     initialize dust routine by setting dust density to zero.
          do 40 j=1,mny
          do 40 i=1,mnx
   40     rha(i,j)=0.0
          do 50 i=1,mnx
          delta(i)=0.02*(ycor(2)-ycor(1))
          deltas(i)=delta(i)/8.0
          delta2(i)=0.0
          theta(i)=delta(i)*7./32.
          mdot(i) = 0.0
          tauw(i) = 0.0
 50       continue
 167        continue
       return
      end
          subroutine duster(istep,time)
c---------------------------------------------------------------------
c     adds dust so as to model the dust pickup behind the shock front.
c     the SOUR model used is based upon the S**3 model. the dust mass
c     added at each step is .08*area*u*rho(air).
c---------------------------------------------------------------------
          parameter (mnx=150, mny=200, mxy=202)
          parameter (mnxx=1, mnyy=1)
c
          integer alpha
          logical lmgrid, lstat, lnuc, therml, lgrav, leos
          real rha(mnx,mny), rhb(mnxx,mnyy), rhc(mnxx,mnyy)
          real tem(mnxx,mnyy), sci(mnxx,mnyy), rho(mnx,mny)
```

```
      real rvx(mnx,mny), rvy(mnx,mny), erg(mnx,mny)
      common nx, ny, nxp, nyp, nspec, idump, idiag, imgrid
      common istat, inuc, therml, igrav, ieos
      common alpha, gamma, gm1, dt, dx, dy, rhomin
      common tem, sc1, rho, rvx, rvy, erg, rha, rhb, rhc
c
      real xcor(mxy), xcoro(mxy), ycor(mxy), ycoro(mxy), unit(mxy)
      common /grids/ xcor, xcoro, ycor, ycoro, unit
c
      real rhom(mxy), pres(mxy), vel(mxy), uh(mxy)
      real ergk(mxy), srvx(mxy), srvy(mxy), serg(mxy)
      real rha0(mxy), rhb0(mxy), rhc0(mxy), vh(mxy)
      real rho0(mxy), rvx0(mxy), rvy0(mxy), erg0(mxy)
      real rhoh(mxy), rvxh(mxy), rvyh(mxy), ergh(mxy)
      real delx(mxy), dely(mxy), scrh(mxy), sgrv(mxy), sgam(mxy)
      real rmin(mxy), rmax(mxy), rvr(mxy)
      common /scrch/ rhom, pres, vel, uh, ergk, srvx, srvy, serg,
     1 rha0, rhb0, rhc0, vh, rho0, rvx0, rvy0, erg0, rhoh, rvxh, rvyh
     2 , ergh, delx, dely, scrh, sgrv, sgam, rmin, rmax, rvr
c
      real pr00(mxy), rh00(mxy), er00(mxy), gp00(mxy), gravy(mxy)
      parameter (jthm=1)
      real fsky(mxy), rhtl(mnx,jthm)
      common /grav/ pr00, rh00, er00, gp00, gravy, fsky, rhtl
     1 , jtherm
c
      real dirt(mxy), rair(mxy), rhae(mxy), eint(mxy), dnew(mxy)
      equivalence (dirt,rhb0), (rair,rha0), (rhae,rhc0), (dnew,srvx)
      equivalence (eint,srvy)
      data con3, ff, edp0 /3.509e-7,0.1,50./
c
c    determine location of shock front. return if shock front has
c    propagated less than 8 zones to the right.
c         do 10 i=1,mnx
c   10    scrh(i)=cvmgt(float(i),0.0,rho(i,1).gt.1.2*rho(mnx,1))
c         ishk=ismax(mnx,scrh,1)
c         sum=ssum(mnx,scrh,1)
c         if (sum.lt.1.0) return
c         inject=max0(1,ishk-8)
         inject=mnx
         if (inject.eq.1) return
c
c    calculate kinetic energy.
      do 20 i=1,inject
      rhom(i)=1.0/rho(i,1)
      rair(i)=rho(i,1)-rha(i,1)
      ergk(i)=0.5*(rvx(i,1)*rvx(i,1)+rvy(i,1)*rvy(i,1))*rhom(i)
   20 eint(i)=erg(i,1)-ergk(i)-rho(i,1)*gp00(2)
c
c    get pressure.
      call eospl (rho,eint,inject,sgam,pres)
c
c    deposit dust in the flow field behind shock front.
      dy12=(ycor(2)-ycor(1))
      do 30 i=1,inject
      dirt(i)=0.08*rair(i)*dt*ff*rhom(i)*abs(rvx(i,1))/dy12
      dnew(i)=rha(i,1)+dirt(i)
      rha(i,1)=cvmgt(dnew(i),rha(i,1),ergk(i).gt.edp0)
      rho(i,1)=rair(i)+rha(i,1)
      erg0(i)=amax1(0.0,eint(i)+1.e7*dirt(i)*con3*pres(i)/rho(i,1))
      erg(i,1)=cvmgt(erg0(i)+rho(i,1)*gp00(2)+ergk(i)/(rhom(i)*rho(i
     1 ,1)),erg(i,1),ergk(i).gt.edp0)
   30 continue
         if(mod(istep-1,100).eq.0) then
```

```
      write(6,78)  istep,time
78    format(1x,'dirt  at step',i4,' and time',1pe12.5)
       write(6,79) (dirt(i),i=11,30)
79     format(1x,1p8e12.5)
          else
          endif
       return
c
       entry dust0
c
c   initialize dust routine by setting dust density to zero.
       do 40 j=1,mny
       do 40 i=1,mnx
   40  rha(i,j)=0.0
    end
```

APPENDIX B

ABSTRACT FOR DNA

DUST SYMPOSIUM

Three-Dimensional Effects
in Dust Clouds- A Numerical Approach

Mark A. Fry
Science Applications International

Three-dimensional numerical simulation codes combining minimal diffusion algorithms which are exceptionally fast on today's supercomputers have been used to investigate nuclear dust cloud scenarios. Results show persistent multi-dimensional effects for long periods of time. Conclusions based solely on two- dimensional(2-D) calculations are misleading and sometimes incorrect. Modeling of high explosive dust cloud scenarios has also been performed and a unique high explosive and numerical simulation program for dust clouds is proposed. When more than one explosion occurs with proximity, multi- dimensional effects become important. Separation distances up to many fireball radii produce highly complex flows which result in asymmetrical dust cloud geometries. Even separation distances less than a fireball radius can influence cloud shape. The need to perform parametric analysis of these effects has led to the development of a very fast non-diffusive 3-D code, FAST3DD with dust entrainment. Pertinent aspects of the physics modeled will be explained. High explosive dust cloud simulation provides a research path when combined with numerical calculations can lead to new understanding of entrainment processes and cloud develoment. Improvements in the understanding of the dust lofting mechanism have been made with implementation of a turbulent of a turbulent boundary layer model for use with in 2 and 3-D codes. Multi-cloud experiments with HE are proposed with direction from 3-D calculations. Spherical charges of 1000 pounds or more that can be remotely detonated and launched into a proper configuration would be used. A sample scenario has been computed and results will be shown.

APPENDIX C

CHEMICAL EXPLOSIVE CHARGE STUDY USING 3-D FCT CODE

Distributed Explosive Charge Array
(DECA)

Annual Report

Science Applications International Corporation
McLean, Virginia

and

Naval Research Laboratory

Sponsor: Defense Advanced Research Projects Agency

Project Officers: Col. Alexander H. Lancaster and Maj. Randy Lundberg

Program Code: 5G10/6G10

Program Element Code: 62702E

Project Name: DECA–Distributed Explosive Charge Array

Period covered: 1 August 1985 - 31 July 1986

Principal Investigators: Dr. Shmuel Eidelman (SAIC) and Dr. Joseph Baum (NRL)

NRL, Code 4040, Tel. (202) 767-3254

Work performed by: Shmuel Eidelman, David L. Book and Joseph Baum

TABLE OF CONTENTS

SECTION PAGE

Introduction

This report describes the results of the one year feasibility study of the Distributed Explosive Charge Array (DECA) concept, as applied to antitank mine neutralization. The DECA concept was proposed as an area weapon which would have the maximum blast effect for a given weight of explosive. The goal of our study was to substantiate this claim, by detailed numerical simulations and blast experiments. In the current study we considered DECA arrays formed by line charges of explosive positioned with selected distances between charges. The simulation of the blast wave propagation from a multitude of finite length explosive charges was done on a CRAY-XMP supercomputer, using a fully 3–Dimensional, time dependent numerical model. Parametrical study of the maximum pressure and impulse dependence on surface density was done for DECA formed from strands of Primacord. The formation and propagation of blast waves from the DECA was studied for arrays suspended above the ground, and lying on the ground, with different spacings of the Iremite or Primacord charges.

Experiments were conducted at SRI International remote test facility at Corral Hollow, California. Experiments were done with full size DECA arrays of Iremite or Primacord explosives, and were designed to follow numerical simulations. The experimental data was used to verify the computer simulations. To illustrate the effectivness of DECA against antitank mines, in two tests M–15 mines were desroyed by DECA arrays, suspended 25cm and 50cm above the ground.

1. Numerical Simulations

For numerical simulation of the blast waves of complex structures generated by DECA explosions, we used FAST3D computer code. This code was developed recently in the Laboratory for Computational Physics at NRL, and it solves unsteady and fully three-dimensional flow problems, with good resolution of the shock waves and contact discontinuities. This code is vectorized and optimized for the most efficient use on the Cray X-MP supercomputer. More details about this code structure and the numerical method used are given in Ref.1. This reference is added as Appendix A to the current report.

The purpose of the current study was to achieve levels of overpressure and impulse loads which cause collapse of the antitank mine casing. According to the BRL study (Ref 2.), published in 1983, pressure levels of above 2000psi and the blast wave impulse of above 950psi·sec will cause rupture of the M–15 mine casing, in the area of the central fuse cavity. This datum was taken as a reference point for defining the range of explosive surface density able to produce the load needed to destroy antitank (AT) mines.

The size of the DECA array was first choosen to be 4m × 4m. This is a relevant size for potential applications, and it allowed us to estimate the magnitude of the edge effects (nonplanar relief of the blast energy at the edges of the DECA charge). Later, it was realized that the edge effects are small at the time when most of the blast load affects the ground, and it was decided to study arrays of 2m × 2m in order to reduce expenses of the experiments, and improve the accuracy of numerical simulations.

In order to assure good resolution of the blast waves in all spatial directions, all the simulations were done on an evenly spaced computational mesh (60 × 50 × 50). The linear geometry of the individual charges was chosen because of the simplicity of its implementation, both in experiments and simulations. Only one fourth of the physical plan was simulated numerically, because of symmetry considerations.

For a typical simulation, at the time $t = 0$ we deposited energy and density (corresponding to levels observed for the explosives under consideration) into cells of the computational domain, located along straight lines and evenly spaced. Then an inital value problem was solved explicitly for the system of three-dimensional Euler equations. The pressure, density, velocities (in X,Y and Z directions) and energy were recorded in the form of binary arrays of values for every grid point in the computational domain. Because of the large size of these arrays, they were recorded only for a given increment of time or integration step (usualy 8 times per simulation). However, all physical parameters were recorded at every integration step for 144 points, located in the most relevant places of the computational domain (including the points corresponding to the location of the pressure transducers in the experiments).

For analysis of the complex three-dimensional flow field developing after a typical DECA explosion, a package of graphic subroutines was developed. The plots produced by these subroutines are demonstrated below, where we discuss the results of our simulations. Here we list the various functions of the graphics routines:

1. Contour plots of X–Y cross sections of the domain;
2. Contour plots of X–Z cross sections of the domain;
3. & 4. The same as 1. & 2. in color;
5. Time history of pressure, impulse, density and temperature (or total pressure);
6. Contour plots of the impulse distribution on the ground;
7. Graphs of pressure and impulse distribution on the ground.

The first set of computations and experiments was done with Iremite explosive. The surface density of the explosive was about 2.6kg/m^2. Figure 1.1 shows a schematic layout of a typical simulation. Initially, the explosive arrays consisted of nine 4m long lines of Primacord spaced 0.5m and placed on the ground. Because of symmetry, in the computational domain we have 4.5 lines (the line at the center of the array is cut in half by the symmetry plane Z–Y) and the lines are

2m long (again, physical lines of explosive are cut in half by the symmetry plane Z–X). In Figures 1.2(a,b,c,d), pressure contours are shown in the X–Z plane for the blast wave propagating after the explosion of a 100lb charge of DECA explosive formed by nine 4m charges of Primacord. The pressure contours are shown for $t = 0.13$ms, $t = 0.30$ms, $t = 0.49$ms, $t = 0.78$ms after the detonation. It is noticeable that in this case, because of the 0.5m space between the charges, the load on the ground is not uniform. Also, the edge effects are significant in this case. In Figures 1.3(a,b,c,d), pressure contours are shown for the X–Y plane (the ground). The maximum pressure in these plots is allocated along the lines; although this maximum is moving along the ground in time (compare Fig. 1.3b and Fig. 1.3c), this is not enough to issure an even load on the ground. In Figures 1.4(a,b,c) the contour plots of the impulse on the ground are shown for $t = 0.25$ms, $t = 0.50$ms, $t = 1.0$ms, after detonation. The maximum impulse achieved in this simulation was $\approx$ 1500psi $\cdot$ ms. These levels were observed directly under the lines of explosives. In the areas between the lines of Primacord, the impulse is around 450psi $\cdot$ ms. These areas of small impulse are $\approx$ 20cm wide. The blast wave contibutes most significantly to the impulse in the first 0.5ms after the detonation.

In Figures 1.5(a,b), graphs of pressure, impulse, temperature and density are shown for the same simulation, as functions of time elapsed after the detonation. (The coordinates of the station where the recording was made are given in the third line of the graph headings.) The station coresponding to Fig. 1.5a is located under the explosive. That leads to very high presssure and impulse. However, for the station located between the lines of explosives shown in Fig. 1.5b, the maximum pressure is only about 1.8kpsi and maximum impulse is 500psi $\cdot$ ms. Initially, the uneven load distribution was not considered a negative factor, since it was assumed that to destroy an AT mine it would be sufficient to produce a very high load on part of the mine . Later, it was learned that this concept could be applied only to AT mines which are activated by large pressure plates, and it would be inefficient for blast resistant mines. For this reason, later in our study we tried to get more

even distribution of the blast wave load on the ground.

Let's consider in more detail the dynamics of the blast wave for DECA. Simultaneous detonation of a Distributed Explosive Charge Array formed by line charges of explosive will at first lead to cylindrical blast waves in the vicinity of the charges. At a later time, cylindrical blast waves from single charges of the array collide and after multiple reflections, a pseodo planar wave would form. Since the main advantage of the DECA concept is the use of planar blast waves to efficiently produce blast loads on the ground, earlier formation of the planar blast wave leads to increased efficiency. At the limit, the distance between the charges could be so large that DECA will not produce a greater blast load than single line charges of the same weight detonated separately. That will take place when collision of the blast waves from two parallel charges of the array will produce lower pressure than the target threshold. At another extreme, a large amount of very closely spaced lines of explosive will be difficult to detonate simultaneously, and if the line is thinner than a critical radius it will not detonate at all. A feasible DECA device would have parameters laying between these two limits, and one of the tasks of our proposed second-year development program will be to determine the range of DECA parameters where this concept is most effective. In our current study, the main task was to show the feasibility of DECA, and to correlate simulations with experimental tests. For this reason, we worked only with two spacial distances between charges: 50cm and 25cm.

In order to smooth the blast load distribution on the ground it was first decided to use different explosives. Iremite is a very "slow" explosive, with a density of only about $0.9 g/cm^3$. Its detonation velocity is 3500m/sec. It is also produced in strands and it could be used is the same way as Primacord. It was also decided to use 10 lines of explosive instead of 9, so the Z–Y symmetry plane would lie between the lines of explosive. That will extend the size of the array to 4.5m × 4m. In Figure 1.6 the results of this simulation are shown in the form of pressure and impulse distribution on the ground. From the graphs in Fig. 1.6 it could be concluded that

substituting explosive from Primacord to Iremite would not improve the blast load distribution.

In order to achieve a more even blast load on the ground, the distance between was reduced from 50cm to 25cm. For the array of 4m × 4m, 25cm spacing between the charges will lead to 17 lines of explosive. The surface density of explosive in this simulation was same as in previouse simulation with Iremite. In Figures 1.7(a.b) and 1.8(a,b) contour plots of pressure, for X–Z and X–Y cross sections corespondingly, are shown for $t = 0.15$ms and $t = 0.35$ms. It is easy to see in these figures that for that case, the planar front forms almost imediatly after detonation and few contour lines close to the ground in Fig. 1.7a indicate that the blast was originated from the line sources. However, more detailed examination of the pressure time history for the point under the line explosive, shown in Figure 1.9a, and for the point between the lines of explosives, shown in Figure 1.9b, demonstrates large variation of blast load on the ground. In Fig. 1.9a pressure reached the same maximum level as in the case of 10 lines of Iremite (see Figure 1.6), but because the lines of explosive are thinner, in the current simulation the maximum impulse is only 930psi·ms. Between the lines the maximum pressure is 4.1kpsi, twice as large as in the previouse case, and the impulse is 580psi · ms.

Another method to achieve an evenly distributed blast load on the ground is to detonate DECA at some distance from the ground. In that case, multiple collisions of the blast waves from neiboring charges and formation of the planar front would begin prior to the start of the reflection from the ground. Because the incident blast wave in this case would be smoother, its reflection from the ground will produce more evenly distributed load. In order to prove this concept we simulated detonation of the DECA aray of 17 line charges of Iremite (length and weight was the same as in previous simulation) suspended 20cm above the ground. In Figures 1.10(a,b,c,d), results for this simulation are given in the form of pressure, impulse, density and temperature time histories for the points located on the ground. All these figures show ≈ 10% variation in pressure and less than

5% variation in impulse. That shows that suspention is the most powerful practical method to achieve even distribution of the blast load on the ground. The maximum pressure observed in simulation of the detonation of 17 lines of Iremite 20cm above the ground is 3250psi and the maximum impulse was 650psi · sec. This blast load is not suficient for the neutralization of an AT mine. In order to increase the blast load we decided to use strands of Primacord explosive, since the Primacord explosive is about 50% more energetic than Iremite.

In Figures 1.11 and 1.12 pressure contours are shown for the X–Z and X–Y cross sections of the blast wave formed by detonation of an 2.25m × 2m DECA device formed of 10 lines of Primacord spaced 25cm apart and suspended 25cm above the ground. Total weight of explosives in that simulation was 30lb. The pressure contours in Fig. 1.11 are shown for t = 0.06ms, t = 0.12ms, and t = 0.27ms. Better grid resolution allows as to follow details of the evolution of the blast wave for this DECA detonation. At t = 0.06ms the blast wave is formed by cylindrical detonations from the parallel line charges, and colisions between the parallel waves have only started. The primary waves from the line charges can be clearly located in Fig.1.11 at time 0.06ms; at that time the front of the blast wave has not reached the ground. At t = 0.12ms the reflection from the ground begins. The reflected blast wave is stronger between the charges, because of colision of the blast waves from two parallel sources. At t = 0.27ms reflection of the blast wave is fully developed. As it could be seen in Fig.1.12 at that time pressure is distributed evenly on the ground. The levels of pressure and impulse could be examined in more details in Figures 1.13(a.b.c.d), where the time histories of pressure, impulse and density are shown for selected points on the ground. The maximum pressures shown in Figs.1.13 are in the range of 7kpsi to 4.5kpsi and the impulse ≈ 800psi · ms. These levels of blast loads should be suficient to destroy an AT mine. Following this conclusion, in our experiments with AT mines we used a DECA charge of the same geometry and surface density as in the last described simulation, and it was suspended 25cm above the ground.

We have studied the dependence of maximum pressure and impulse, produced by the blast wave on the ground, on surface density of Primacord explosive. In Figure 1.14 the results of this study are shown on the log(Pressure) vs. log(Impulse) graph. DECA charges for this study were about 4m × 4m in size, suspended 25cm above the ground. On the same graph, vulnerability limits are shown for mines and other military targets and graphs representing dependence of M–58 line charge and FAE on surface density (volume density for FAE) are shown. From Fig.1.14 it is possible to conclude that DECA gives distinct advantages over M–58 and FAE. The a dvantage of DECA over FAE is in higher pressure. According to Fig.1.14, DECA is more efficient than M–58, since it can produce the same pressure on the ground using about fourth of M–58 weight. Also, the impulse produced by DECA is larger than M–58's impulse.

2. EXPERIMENTAL

All the experimental work reported here was done by SRI International of Menlo Park, California. In this part of our report, we will use the data and information form Ref.3.

The main objectives of the experimental work described in this report were to provide data for correlation with the numerical simulations, and to perform tests on inert mines that confirm the feasibility of this approach to damage or destroy antitank mines.

Approach and Test Procedures

The experimental technique, to meet the objectives stated above, was to detonate arrays of explosive placed on or above the ground and to measure the airblast overpressure at several points above and to the side of the array. Figure 2.1 shows a schematic layout of a typical experiment. Initially, the explosive arrays consisted of ten 4.5m lines of Iremite spaced 0.5m apart and placed on the ground covering an area 4.5m × 4.5m (Figure 2.2). Each line, was made of two or three strands of 1.125 inch diameter Iremite. Each Iremite strand weighs about 0.59kg/m (1.18lb/ft) and the total charge weight was from 52.7kg (116lb) to 79.1kg (173.9lb). The surface density of the explosive arrays was from 2.6kg/m^2 (0.53lb/ft^2) to 3.9kg/m^2 (0.80lb/ft^2). The lines of Iremite were each initiated by a single 3.65m long (12ft) strand of 25gr/ft Primacord. These lead-in strands were bundled together at a single point and initiated by a single 9.1m long (30ft) strand of Primacord. This initiation scheme produces a detonation wave in each line. The detonation front from each line sweeps along the array simultaneously. The detonation velocity of the Iremite is 3500m/s (11,500ft/s). The entire lead-in and array is detonated within 3.1ms of initiation.

For tests 1 through 6, the Iremite was placed directly on a prepared soil bed. The soil bed consisted of a firm soil base that had been leveled. A thin layer (about

5cm) of sand was placed over the soil and packed in place using a hand tamper. The sand provided a smooth surface on which to lay the Iremite. In later tests (test 7), the Iremite array was suspended 0.25 m (10 inches) above the test bed. Columns of styrofoam were used at 4 points along the length of each line to ensure a known uniform distance from the explosive to the ground.

For tests 8, 9, and 10, the Iremite was replaced by bundles of 400gr/ft Primacord. The spacing between lines of explosive was reduced from 0.5m to 0.25m and the array dimensions were reduced to 2.5m × 2.5m. However, the total explosive surface density (2.6kg/m^2) was approximately the same. Therefore, each line of explosive consisted of 7 strands of 400gr/ft Primacord. Each line had a lineal density of 0.59kg/m (the same lineal density of a single strand of Iremite). With these Primacord lines on 0.25m spacings, the surface density of the array is 2.4kg/m^2 compared with 2.6kg/m^2 for the Iremite array.

The Primacord arrays were suspended 0.25m above the ground (tests 8 and 9) or 0.5m above the ground (test 10) in the same manner as the Iremite array in test 7. The Primacord arrays were initiated in the same way as the Iremite arrays.

Instrumentation for these tests consisted of up to 6 pressure gauges mounted in 3 steel poles (Figure 2.2) that were buried in the ground in the central area of the test bed. The poles were 10.2cm diameter (4in) pipes approximately 4.8m long (15.5ft) long that were buried 1.5m (5ft) leaving 3.3m (10.5ft) above the ground. Two of the poles were 1m apart and third pole was 2.25m to one side (Figure 3). Gauge adaptors were inserted in each pole. Gauges could be placed 1, 2, or 3m above the ground in each pole. For the tests described here, only gauge positions 1m and 2m above the ground were used. For tests 1, 2, and 3, piezoelectric pressure gauges (PCB Model 119) were used. Because of large transverse accelerations in the steel poles, these gauges produced unacceptable records. For tests 4 through 10, piezoresistive pressure gauges (Endevco Model 8530A) were used. These gauges produced fully acceptable records. The frequency response of these gauges is about 90 kHz, which means that measurements of pressure rise times of about 2μsec to

$3 \mu s$ are possible. Figure 2.3 shows a layout of the array for Tests 4 and 5 showing gauge positions and gauge numbers, and the explosive positioning.

Pressure data were recorded on a high speed tape recorder that has a frequency response of 80 kHz. The analog tape was played back through a recording digital oscilloscope. The digitized data were stored on a floppy disk for later reduction by a computer to produce hard copy records.

Inert antitank mines were exposed to the blast in tests 8, 9, and 10. These last three tests were designed to establish whether such an explosive array could disable an antitank mine. The inert mines consisted of a mine body that was filled with candle wax to simulate the mine explosive (Figure 2.4a). Before each test, an inert booster and detonator were placed in the pressure plate assembly on the top of the mine and the arming cap was screwed into place.

On these tests, the explosive arrays were suspended either 0.25m (tests 8 and 9) or 0.5m (test 10) above the ground (Figure 2.4b). The antitank mine was placed in the center of the array. The explosion array was positioned so that the instrumentation pole containing gauges G5 and G8 was one mine diameter from the edge of the center mine ($\approx 30cm$). The other instrumentation pole containing gauges G4 and G7 was on the edge of the small array. The third pole was not instrumented. The mine was placed on the hard subsoil, about 5cm (2in) below the sand bed surface, and sand was packed around the mine.

Test Results

Table 1 summarizes the results of the 10 tests performed in this program. As mentioned earlier, tests 1, 2, and 3 produced no usable data, although these tests did demonstrate that the instrumentation poles and tests bed were able to withstand repeat testing. Tests 4 and 5 demonstrated the reproducibility of the technique. Figure 2.5 shows records from these two tests. Other than differences in the peak pressure of up to 35 %, the records are similar. The differences in the peak pressure can be attributed to the spacial gradient of the wave fronts as they pass the fixed gauge position. Small changes in positioning of the DECA explosives

on the ground could lead to a change in the effective gauge position. and that in turn could lead to a change in pressure record, because of the large spacial gradient of the blast wave pressure. Pressure gauges G7 and G8 are located symmetrically, and should produce similar pressure records. However, gauges G7 and G8 recorded substantially different levels of pressure. We will try to explain this difference in the next part of the report.

On test 6, the charge density was increased by 50% from 2.6kg/m^2 to 3.9kg/m^2 The pressure measured at the gauge 7 location 1m above the ground was 3964kPa, which is an increase of 31% from the pressure measured in test 5. An increase of 50% in pressure should not be expected in this case, since pressure is linearly proportional to blast energy only for a planar blast wave. In our case, at the distance of 1m above the ground, pressure is some complex function of energy in between the laws for planar and cylindrical blast waves. The increase in pressure is even smaller for gauges G4 and G5 for an unclear reason. More detailed measurements are needed in order to study the dynamics of the blast wave from DECA, and its dependence on the energy and geometry of the array.

In test 7, the nominal explosive array was suspended 0.25m above the ground. The effect of this configuration was to decrease peak pressure from about 3000kPa (435psi) to about 2860kPa (415psi) at the 1m point, and from about 2500kPa (363psi) to about 2240kPa (325psi) at the 2m point (Figure 2.6). The rise time of the pressure pulse was also increased when the explosive was suspended above the ground.

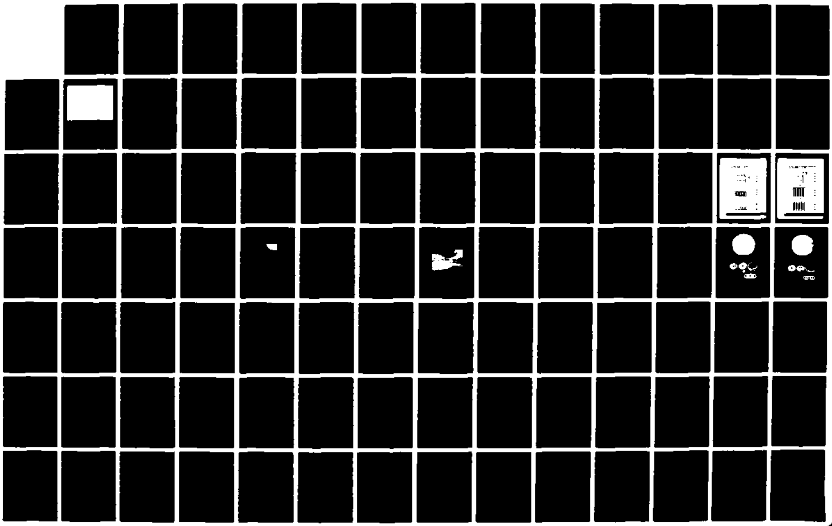
For tests 8, 9, and 10, the Iremite was replaced by bundels of 400gr/ft Primacord. The spacing between lines of explosive was reduced from 0.5m to 0.25m and the array dimentions were reduced to 2.5m × 2.25m. However, the average surface density of explosive ($2.6kg/m^2$) was maintained. The Primacord explosive (PETN) is about 50% more energetic than the Iremite explosive, so for the same surface density, a larger pressure is expected from a Primacord array than from an Iremite array. This proved to be the case, as evidenced by comparing the results of tests 7

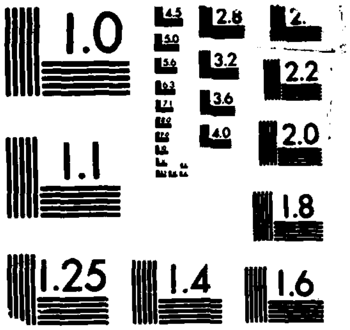
and 9. Test 7 used Iremite suspended 0.25m above the ground, whereas test 9 used the same areal density of Primacord suspended 0.25m above the ground. The peak pressure at the 1m point was 2860kPa (415psi) for the Iremite array and 4860 kPa (705 psi) for the Primacord array, a 70% increase in pressure. The large increase in pressure proves that for a smaller spacing of the line charges the planar wave will form and that will lead to higher overpressures per given weight of explosives. Also included in tests 9 and 10 were inert antitank mines. These mines included a mine casing that was filled with candle wax to simulate the mine explosive, a pressure plate assembly, and an inert booster and detonator. The mines were about 30cm (11.8in.) in diameter and about 15cm (6in.) high. They were placed in the center of the Primacord array, which was 25cm about the ground surface. The mines were centered between lines of primacord that were spaced 25cm apart. The mines were buried so that their bases were on the firm soil subgrade about 5cm (2in.) below the sand bed surface. The upper 10cm (4in.) of the mine was above the ground level.

On test 9, the mine was severly crushed and the pressure plate, the detonator, and the booster assemblies were completely blown off of the mine. Figure 2.7 shows the damage to this mine. On test 10, the explosive array was raised to 50cm above the ground. On this test, the mine was crushed similarly to that in test 9, and the upper portion of the pressure plate and the detonator were blown away from the mine. The lower pressure plate and the rubber seal around the pressure plate along with the booster remained on the mine. This damage can be seen in Figure 2.8a. When the mine was examined after the test, the booster and lower pressure plate assembly were easily removed from the mine. The pressure plate pieces, the detonator, and the booster are shown in figure 2.8b.

AD-A193 152 NUMERRCAL MODELING OF AIRBLAST(U) SCIENCE APPLICATIONS 2/3
INTERNATIONAL CORP MCLEAN VA M A FRY JUN 87
SAIC-87/1701 N00014-86-C-2197

UNCLASSIFIED F/G 19/11 NN





MICROCOPY RESOLUTION TEST CHART
UREAU OF STANDARDS-1963-A

3. Comparison Between Numerical Simulations and Experiments

One of the objectives of the current study was to compare results of numerical simulation of DECA to blast experiments with DECA charges. Because of budgetary constrains we could compare only pressure history in locations above the DECA charge, where pressure has been measured. Only in 6 experiments of the 10, were useful pressure measurments made, and data scatter for identical experiments is in a 30% range. In this environment, only limited number of valid comparisons could be made, and in this section we will discuss them.

First, simulation corresponding to experiments 4 and 5 will be compared with experimental data. In Figure 3.1, measured in test 4 and simulated pressure histories are shown for the location of gauge G7. The simulated pressure closely repeats the patern of the blast wave observed in the experiment. In this case the maximum pressure levels are predicted within 10% of the experimental values. The location of first, second and fourth pressure peaks are predicted accurately. However the third peak is missing in the experimental data. Study of the pressure records at points of the computational domain adjacent to the point G7 shows that at a distance of 10cm from the point G7, the third peak disapears. So, slight inaccuracies in the alignment of the explosive charges on the ground could produce the observed discrepancies in the pressure time–history graph.

Gauges G7 and G8 are symmetrically located relative to the vertical cross section of the DECA charge, and because we were modeling only assymetric part of the array, both of these gauges will correspond only to one point in the computational domain. In Figure 3.2 we compare the pressure record of the gauge G8 obtained in test 4 with numerically simulated pressure for the same point. The character of the graphs reperesenting calculation and experiment are similar; however, simulation predicts $\approx$ 70% higher peak pressure. The difference between the simulation and pressure record obtained by gauge G8 is the same as between gauges G7 and G8, which should show the same pressure, since they located symmetricaly. One of the reasons for this discrepancy could be the obliqueness of the blast wave in

the experiment. The tests 4 and 5 was done with Iremit which has relativly low detonation velocity of 3500m/sec At that velocity, the detonation wave will reach the center of the DECA array in ≈ 0.57ms. That is comparable to ≈ 0.42ms which takes to the blast wave to travel from the ground to the location of the gauge G7 (or G8). That means that at the time the blast wave will reach the pressure transducer it will be oblique to horisontal plane at 36 deg angle. So, small deviation in the orientation of gauges G7 and G8 can lead to large differences in measured pressure. At the same time numerical solution currently do not simulate runing detonation wave (that was proposed in our next years program) and its assumes that array detonates simultaneously. As we will see below in case of 2m $\times$ 2m Primacord array the obliqueness of the blast front is smaller and the described effect would be negligable. However, we think that more experimental and computational study is needed to assess in details formation of the oblique blast fronts and its effect on the ground blast load. That subject is specialy important for design of the initiation schemes for the DECA charges. Test 5 was an exact repeat of test 4 and results of pressure measurment in this test are very similar to results of test 4. However, the large difference in pressure measured in gauges G7 and G8 reported for test 4 was also observed in test 5. That means that this discrepancy is systematic.

In test 9, 10 lines of Primacord explosive was used. Spacing between the lines was 25cm. In Figure 3.3 we compare the pressure history measured by gauge G8 in test 9 and numericaly simulated results for the same point. For that point, the results of simulation and experiment are very close, although the simulation predicted a stronger than measured rarefraction wave behind the first shock wave. In Figure 3.4, simulation and experiment are compared at the point where the G7 gauge is located. Please note that for this experiment, gauges G8 and G7 are not located in symmetrical poisitions. As is shown in Fig. 3.4, gauge G8 is located 75cm above the center of the array and gauge G7 is located at the same height above the array's edge. In Fig. 3.4, the experimentally measured maximum pressure is about 30% larger than simulated. Also, simulation predicts two shock

fronts, and the experiment measured only one front for the G7 gauge. The reasons of these discrepancies are unclear, and we point again to the inaccuracies of laying the Primacord strands and of placing the gauge. More experiments and calculations are needed in order to provide a better data base for comparision. We believe that the qualitative results of simulation for gauge G7 are correct, since at least two shock fronts should be present as a result of colisions between the blast waves from adjacent parallel lines, equidistant from the point of measurment.

In Figures 3.5 and 3.6, results for simulation and experiment are compared for the points where gauges G4 and G5 were located. Both gauges were located at a distance of 1.75m above the ground. At that distance, as a result of the multiple colisions between the shock waves, the blast wave propagates with one front. In Figures 3.5 and 3.6, results of experimentally measured pressure and calculation are in the range of expected accuracy for experiments and calculations (±15%).

4. Summary and Conclusions

As a result of the one-year, DARPA supported (program code 5G10/6G10), exploratory effort to investigate the Distributed Explosive Charge Array (DECA) we come to the following main conclusions:

a. Both experiments and computer simulations point to substantial advantages of the DECA concept over known mine clearing methods.

b. The computer simulations have been found to be in general agreement with the experimental data, although the accuracy of the experiments should be improved. In particular, the ringing and heating of the pressure gauges should be reduced. Future experiments should be better instrumented, in order to provide a clearer picture of the blast propagation. Also a larger experimental data base is needed to clarify some inconsistencies in pressure records (such as differences in data from gauges G7 and G8 in experiments 4 and 5). Measurments on the ground will add very important information about pecularities of the reflection of the blast wave created by DECA.

c. Computer simulation indicated that 120 lb of DECA explosive will produce targeted impulse and overpressure on the ground ($I \approx 1\text{psi} \cdot \text{sec}, P \approx 4000\text{psi}$). This impulse and pressure is sufficient for AT mine destruction, and will clear mines over a 16m^2 area. That constitutes an improvement by more than a factor of five over the closest competetive device (M-58 line charge).

d. The ability of DECA to destroy AT mines was verified experimentally. In two experiments DECA charges were suspended at 25 cm and 50 cm above the ground. In both of these experiments AT mines lying on the ground were destroyed. This shows that relatively large variations in the distance from the target did not affect substantially the ability of the charge to destroy mines. This also indicates that DECA will not need to be deployed with high precision, which will make its production cost-effective. However, the threshold at which such mines can be

destoryed needs to be determined. The effects of burying the mine below ground or the effects of reducing charge density (lower charge weight, increased charge spacing. increased charge standoff, or less energetic explosive) are all parameters that need to be investigated to establish mine lethality for near surface explosive arrays.

References

1. M. Fry, P. Kamath and D. Book, *Three-Dimensional Algoruthm Desing for Hydrodynamic Calculations*, International Symposium on Computational Fluid Dynamics, Tokyo, 1985.

2. F.H. Gregory and A.D. Gupta, *The Use of ADINA for Analysis of Mines with Explosive Fills*, Computers & Structures, Vol 17, No. 5–6, pp. 625–633, 1983.

3. C.M. Romander, *Measurment of Airblast and Responce of Antitank Mines from a Planar Explosive Array*, Iterim Report, SRI–International, May 1986.

Figures and Table Captions

Figure 1.1 Schematic layout of a typical simulation.

Figure 1.2a Pressure contours for a 9 lines $4m \times 4m$ DECA array test. X–Z cross section.

Figure 1.2b Pressure contours for a 9 lines $4m \times 4m$ DECA array test. X–Z cross section.

Figure 1.2c Pressure contours for a 9 lines $4m \times 4m$ DECA array test. X–Z cross section.

Figure 1.2d Pressure contours for a 9 lines $4m \times 4m$ DECA array test. X–Z cross section.

Figure 1.3a Pressure contours for a 9 lines $4m \times 4m$ DECA array test. X–Y cross section.

Figure 1.3b Pressure contours for a 9 lines $4m \times 4m$ DECA array test. X–Y cross section.

Figure 1.3c Pressure contours for a 9 lines $4m \times 4m$ DECA array test. X–Y cross section.

Figure 1.3d Pressure contours for a 9 lines $4m \times 4m$ DECA array test. X–Y cross section.

Figure 1.4a Impulse on the ground at $t = 0.25$msec.

Figure 1.4b Impulse on the ground at $t = 0.50$msec.

Figure 1.4c Impulse on the ground at $t = 1.00$msec.

Figure 1.5a Time history of pressure, impulse, temperature and density on the ground under the explosive.

Figure 1.5b Time history of pressure, impulse, temperature and density on the ground between lines of explosive.

b . Computer simulation.

Figure 3.5 Comparison of data from gauge G4 (test 9) and simulation.

a . Experimental data

b . Computer simulation.

Figure 3.6 Comparison of data from gauge G5 (test 9) and simulation.

a . Experimental data

b . Computer simulation.

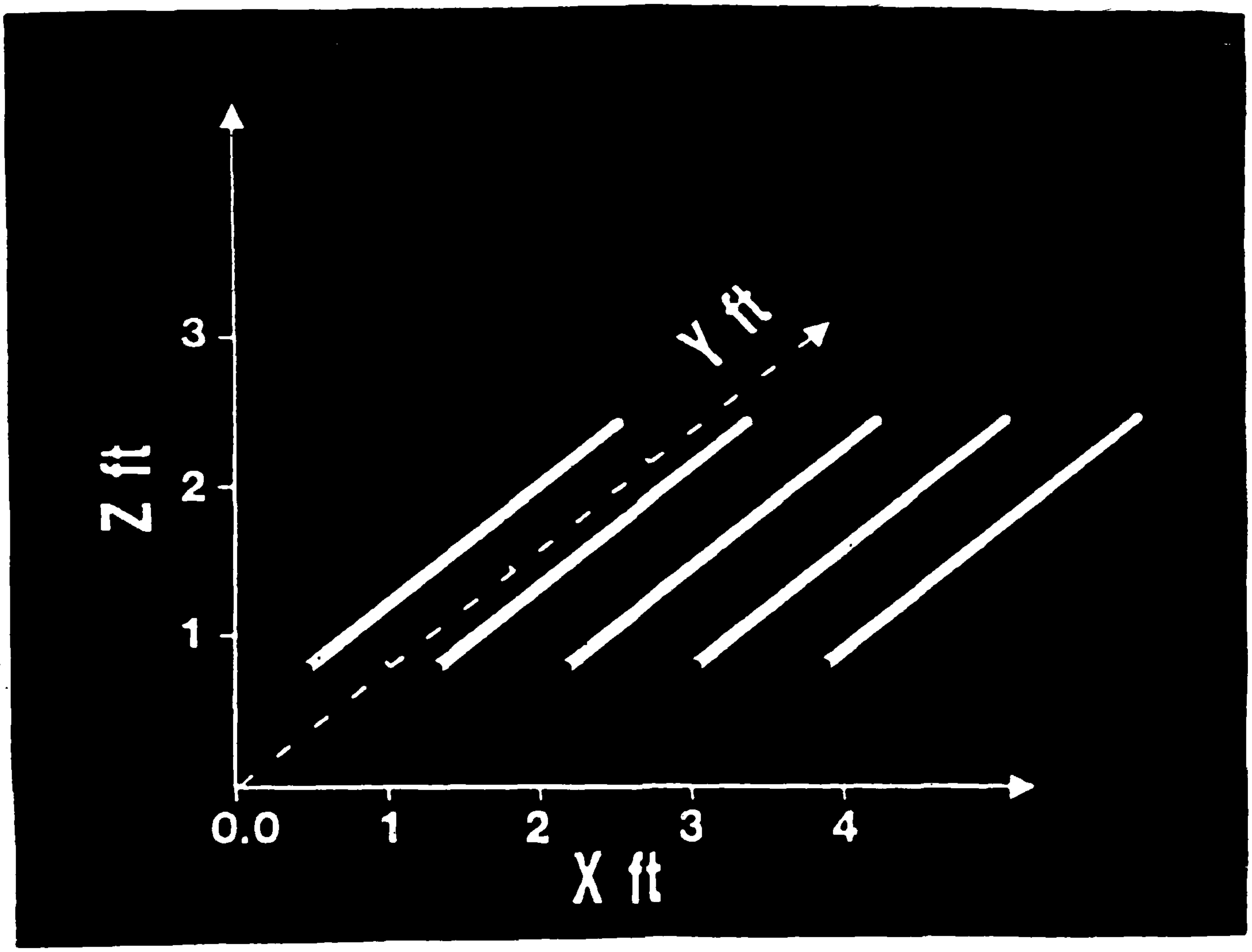


Figure 1.1 Schematic layout of a typical simulation.

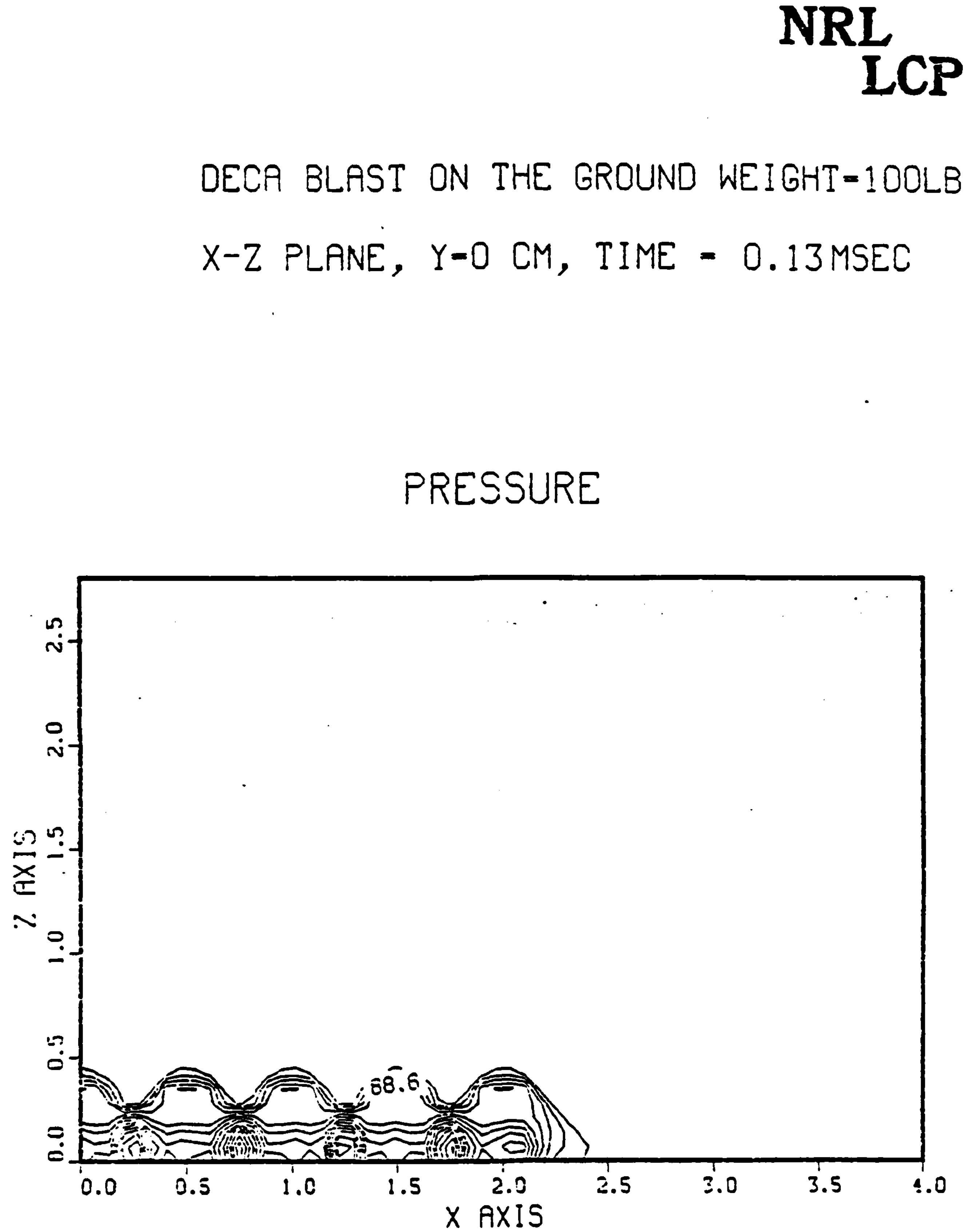


Figure 1.2a Pressure contours for a 9 lines $4m \times 4m$ DECA array test. X-Z cross section.

NRL
LCP

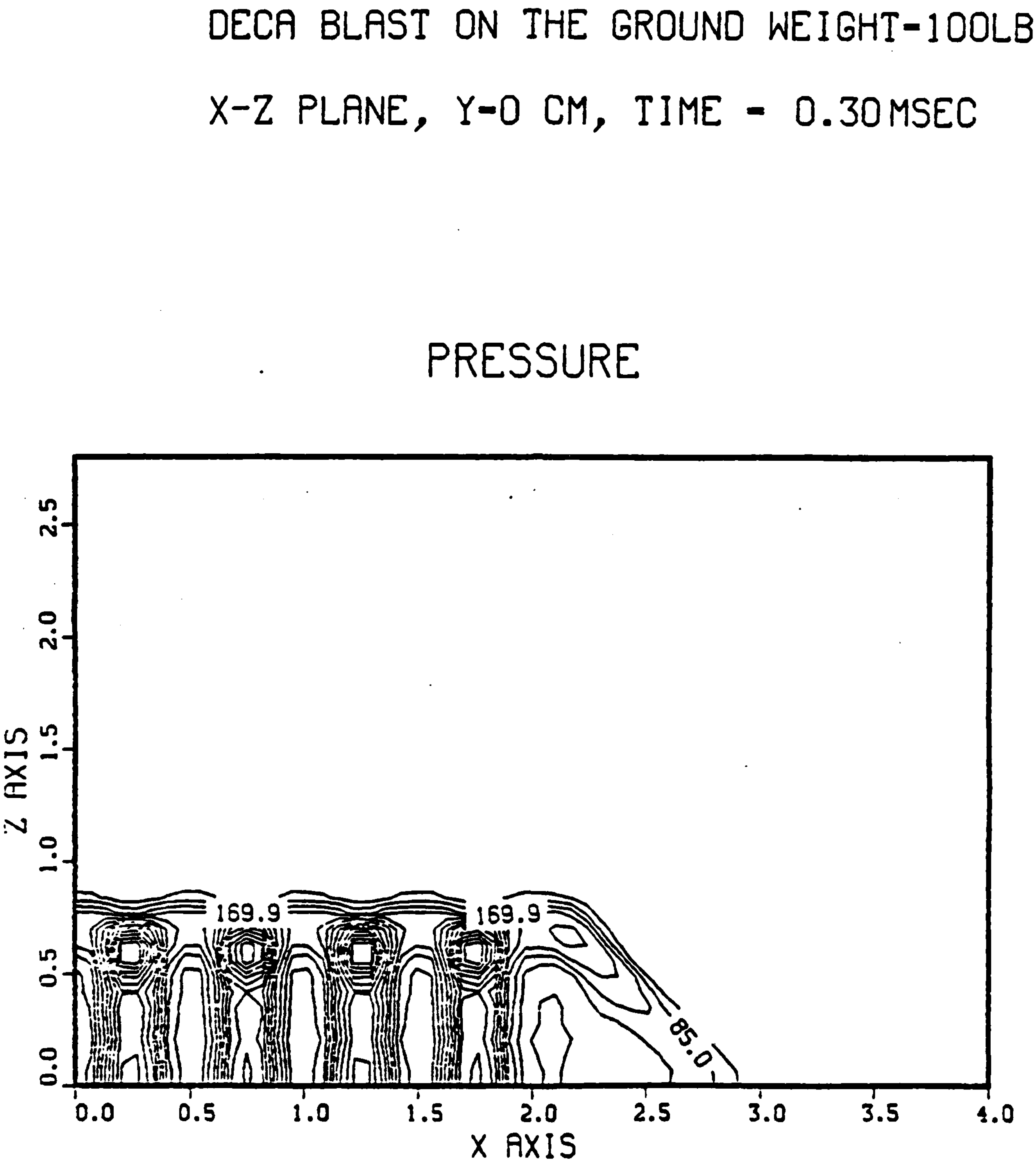


Figure 1.2b Pressure contours for a 9 lines $4m \times 4m$ DECA array test. X–Z cross section.

NRL
LCP

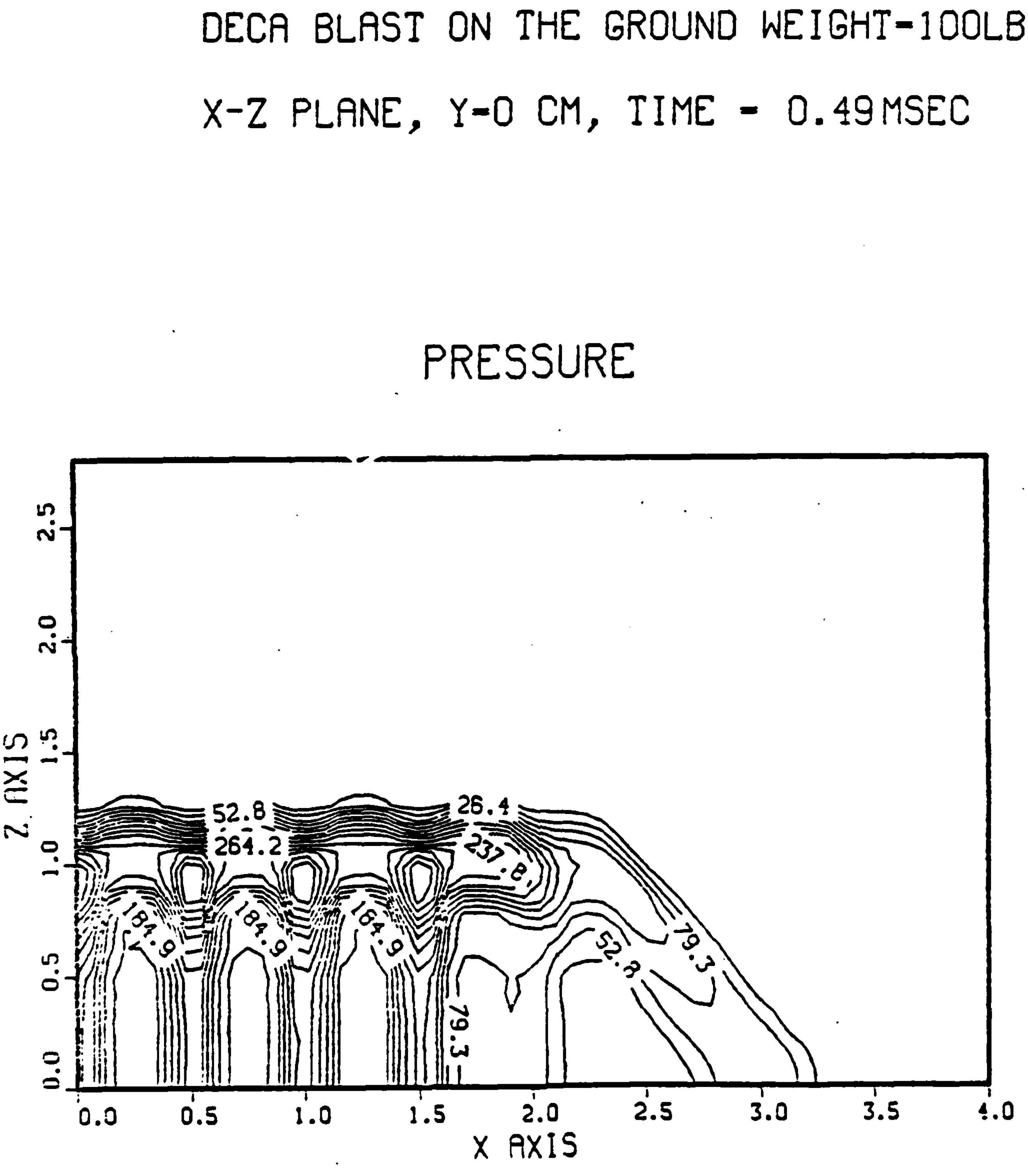


Figure 1.2c Pressure contours for a 9 lines $4m \times 4m$ DECA array test. X–Z cross section.

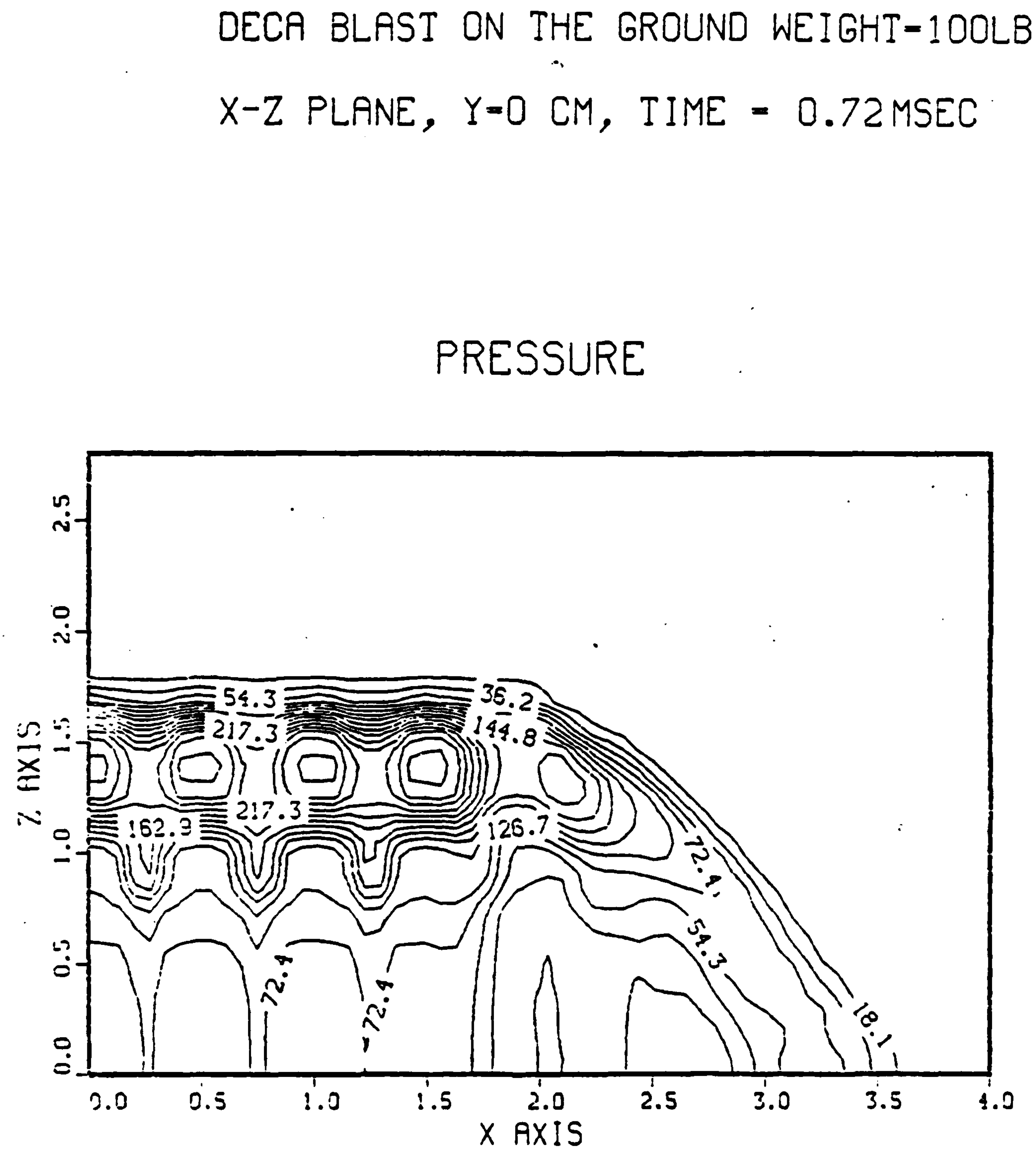


Figure 1.2d Pressure contours for a 9 lines $4m \times 4m$ DECA array test. X–Z cross section.

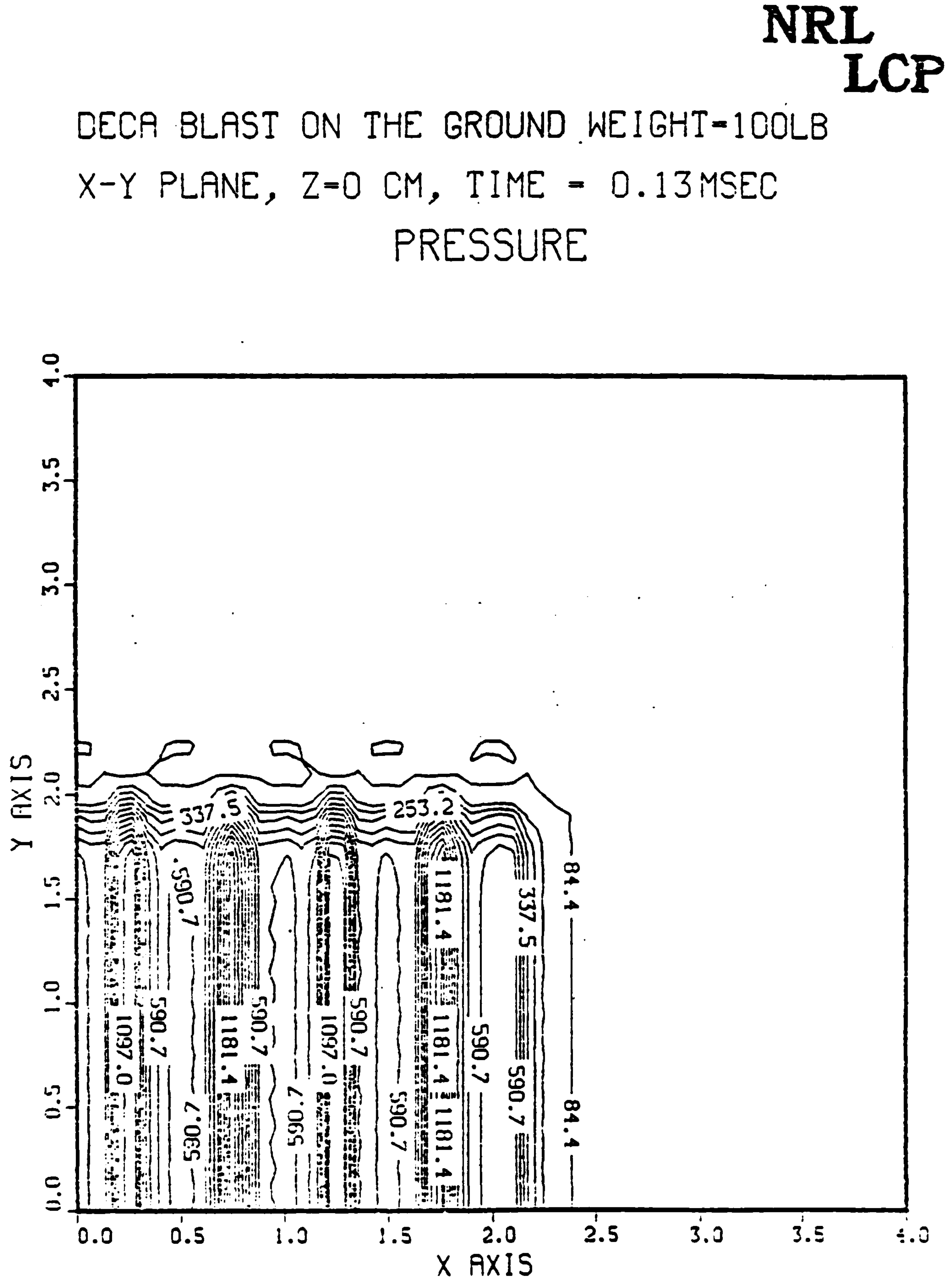


Figure 1.3a Pressure contours for a 9 lines $4m \times 4m$ DECA array test. X-Y cross section.

NRL
LCP

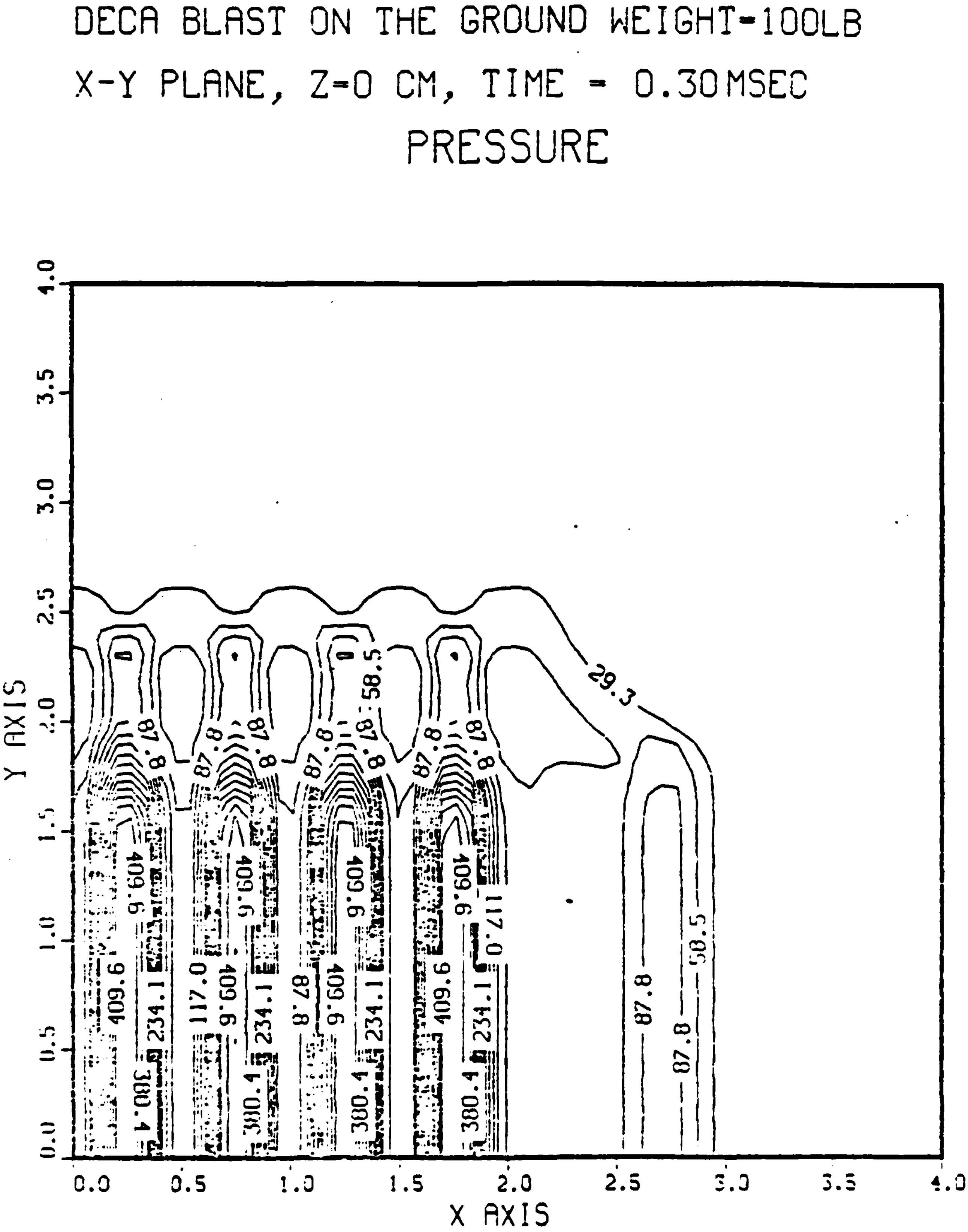


Figure 1.3b Pressure contours for a 9 lines $4m \times 4m$ DECA array test. X-Y cross section.

NRL
LCP

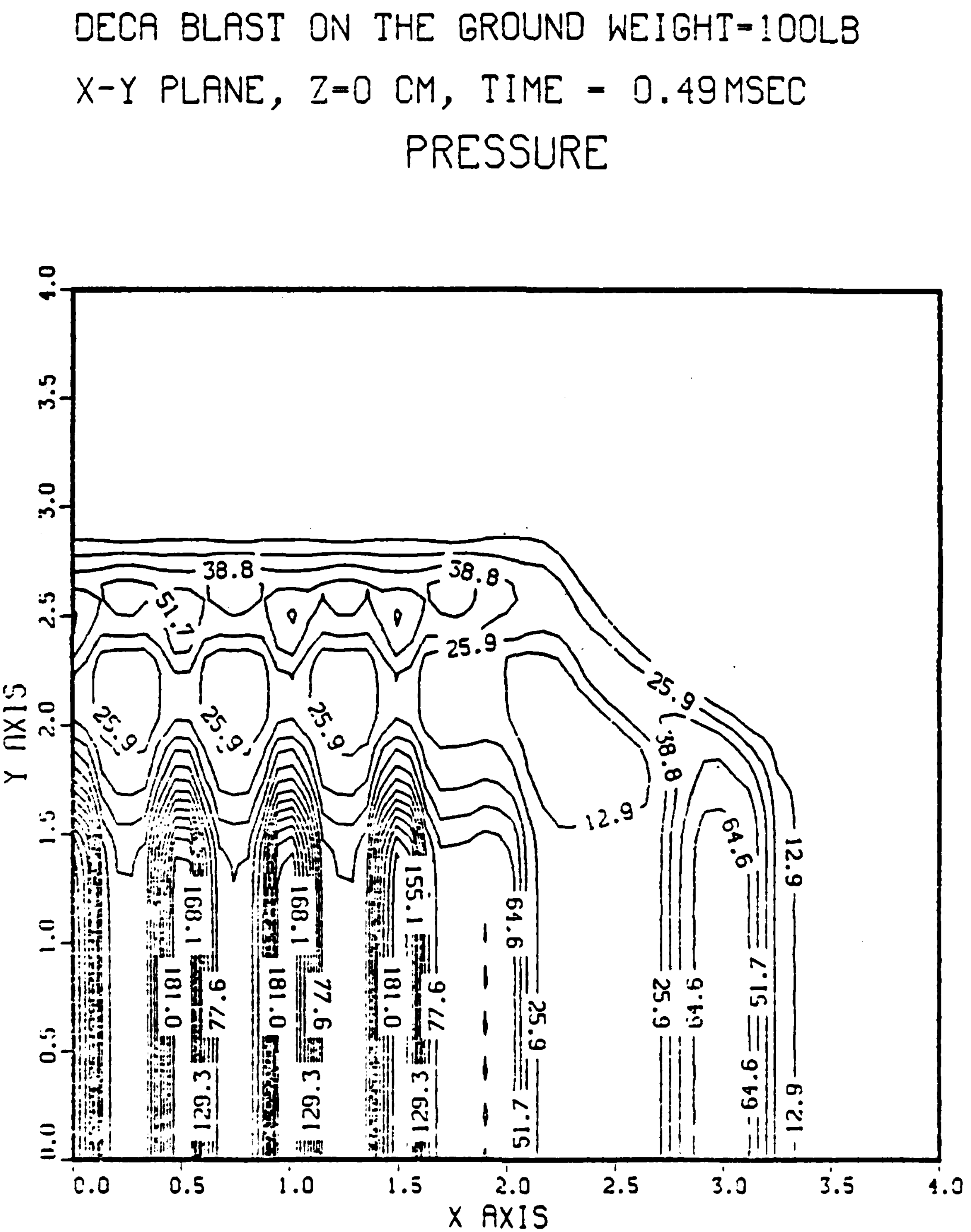


Figure 1.3c Pressure contours for a 9 lines $4m \times 4m$ DECA array test. X–Y cross section.

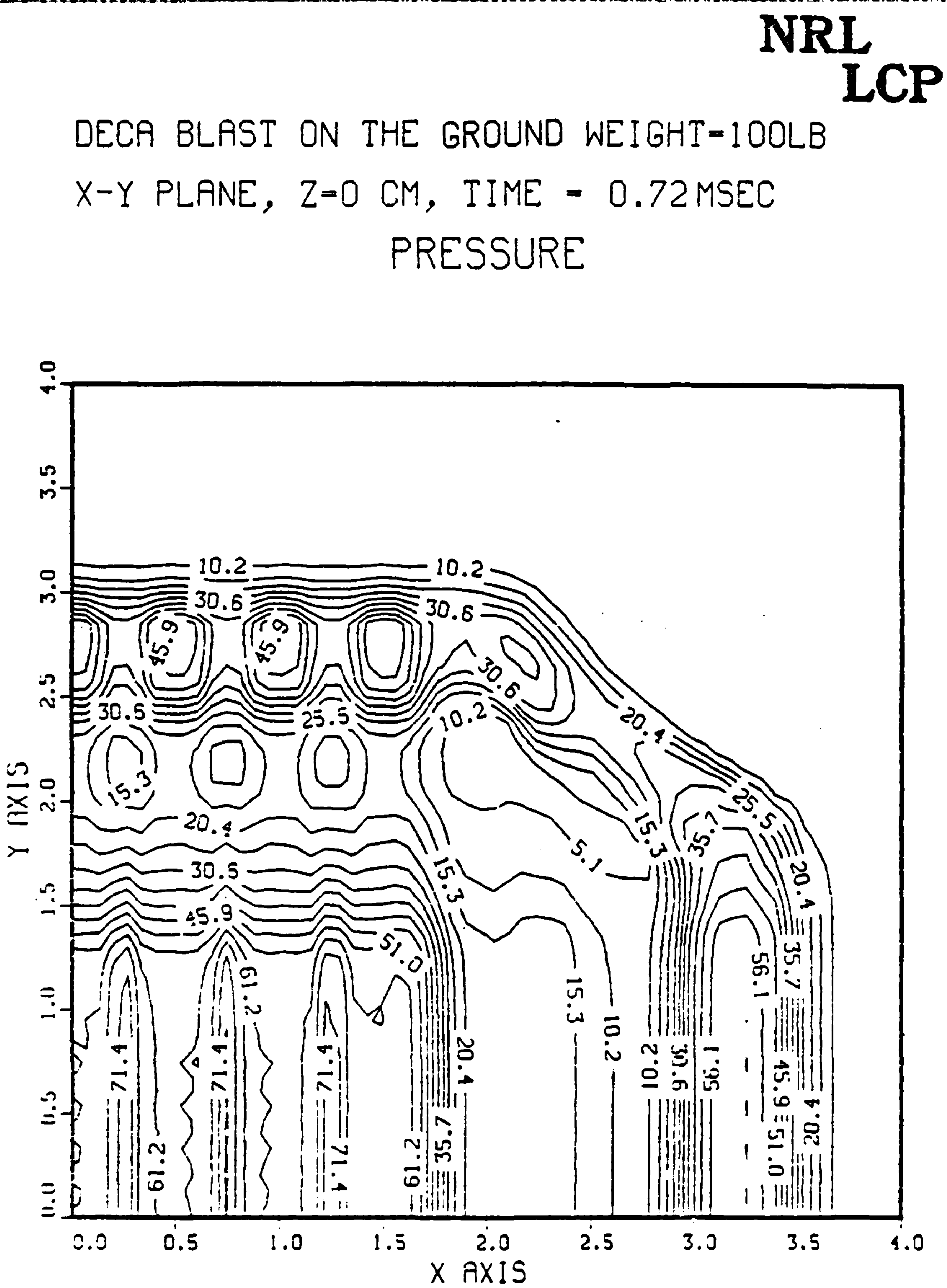


Figure 1.3d Pressure contours for a 9 lines $4m \times 4m$ DECA array test. X-Y cross section.

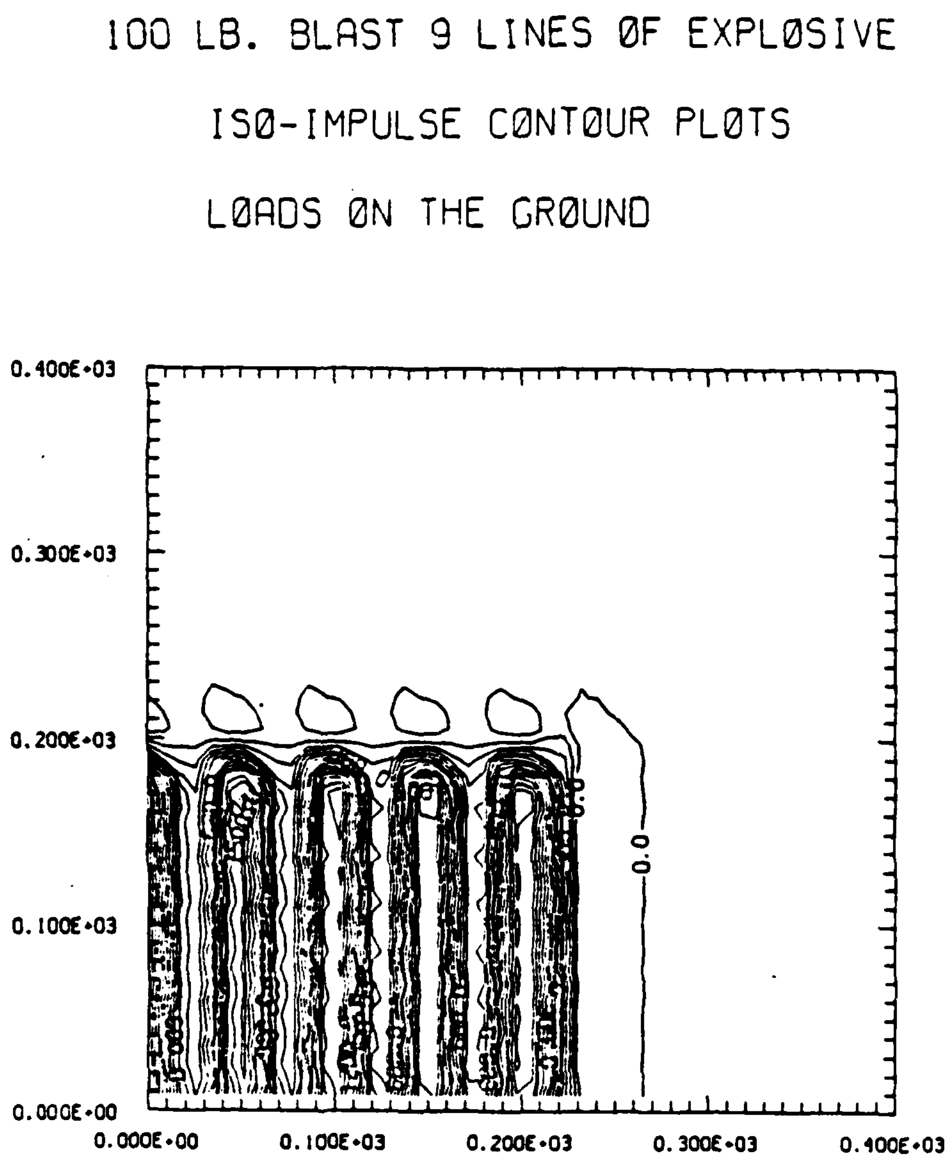


Figure 1.4a Impulse on the ground at $t = 0.25$msec.

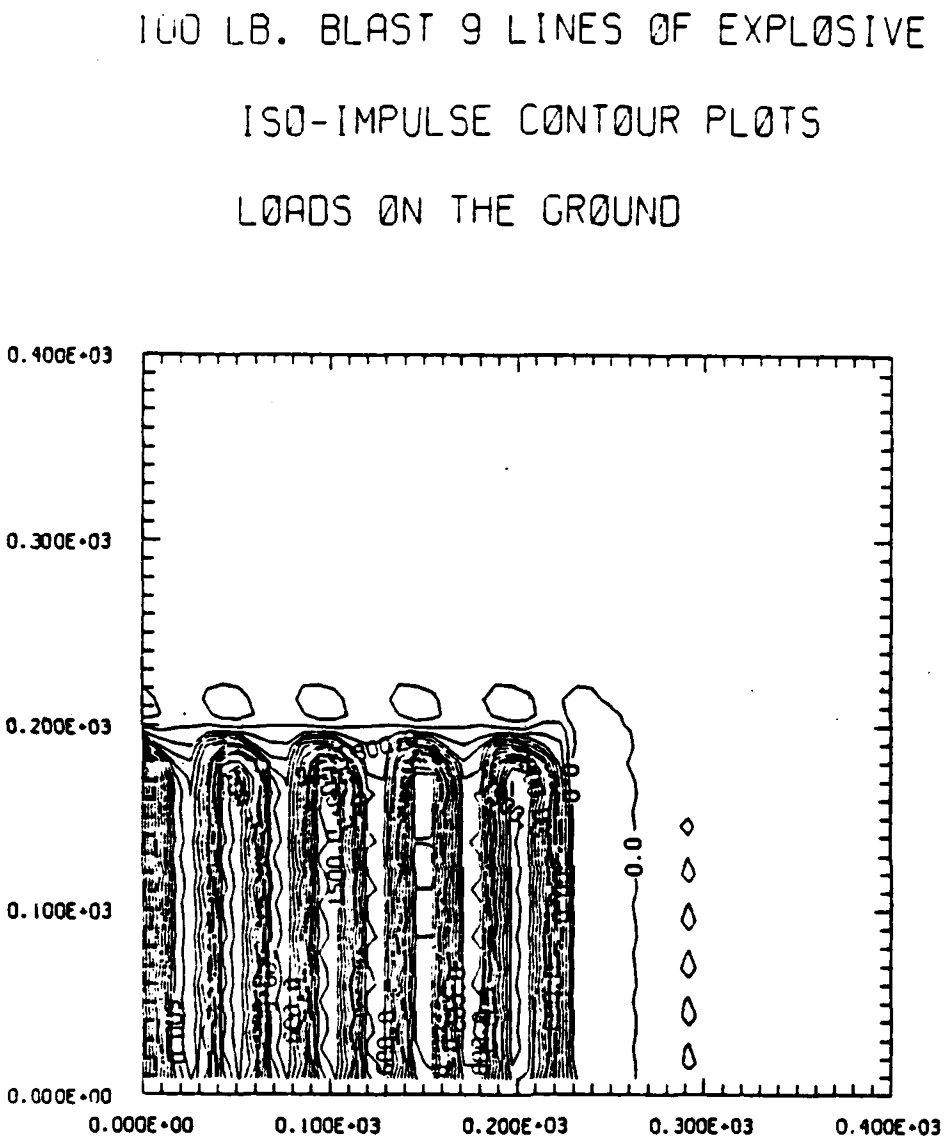


Figure 1.4b Impulse on the ground at $t = 0.50$msec.

100 LB. BLAST 9 LINES OF EXPLOSIVE

ISO-IMPULSE CONTOUR PLOTS

LOADS ON THE GROUND

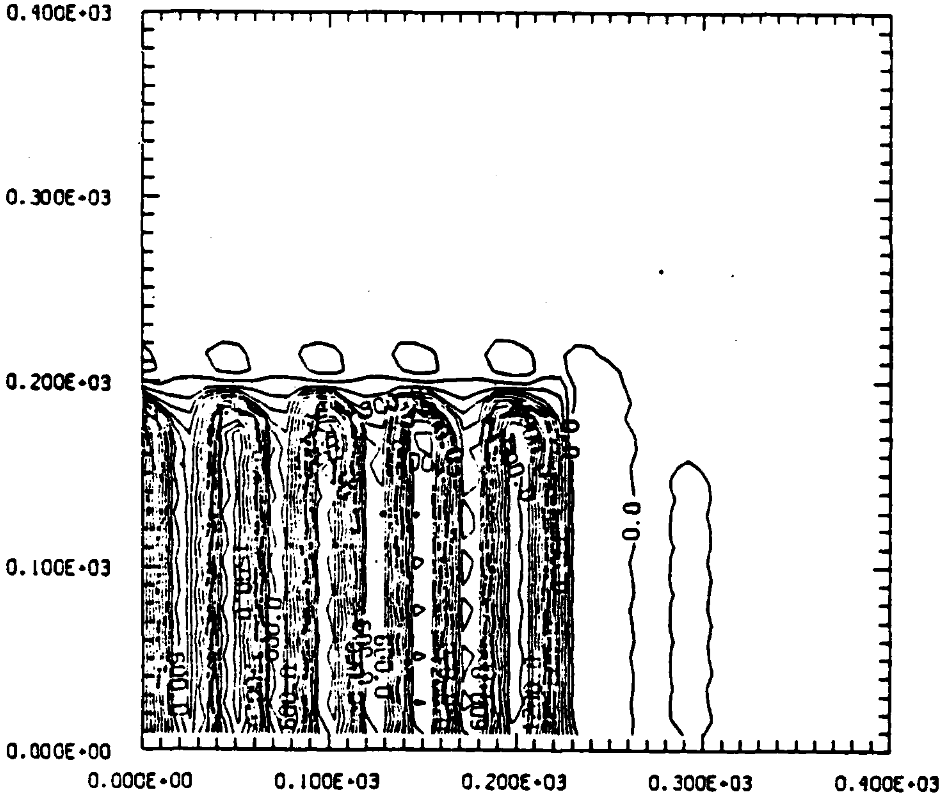


Figure 1.4c Impulse on the ground at t = 1.00msec.

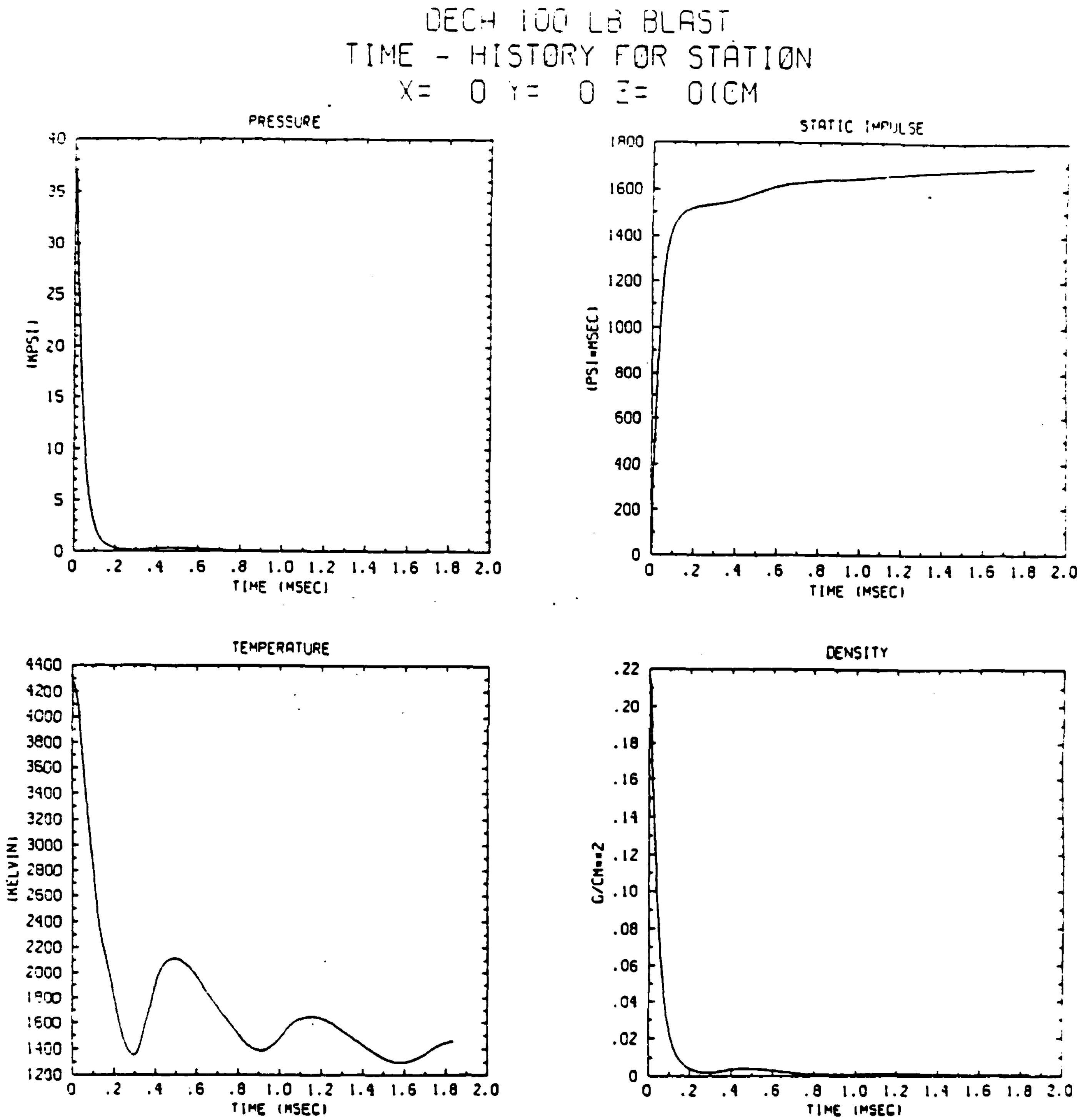


Figure 1.5a Time history of pressure, impulse, temperature and density on the ground under the explosive.

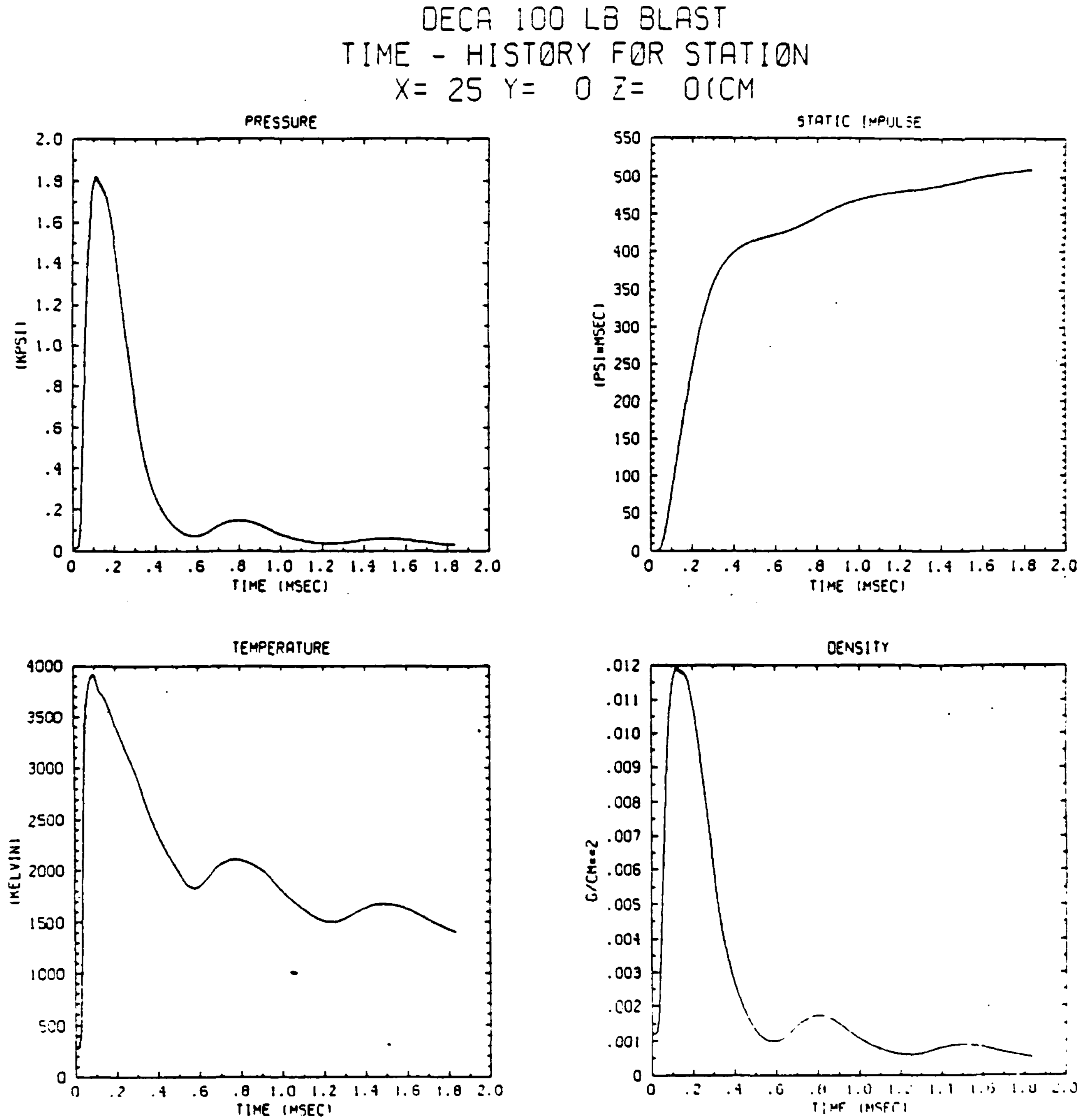


Figure 1.5b Time history of pressure, impulse, temperature and density on the ground between lines of explosive.

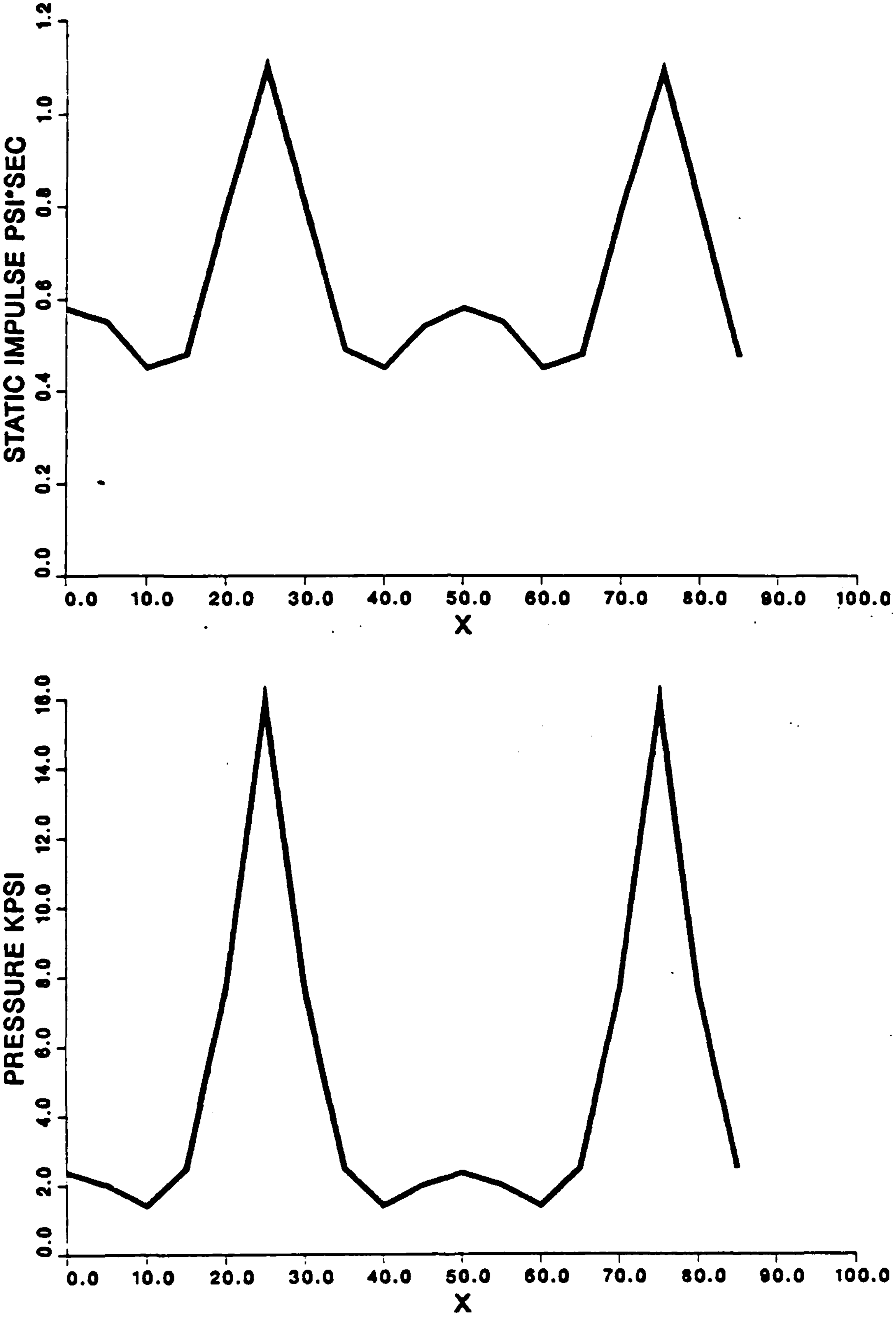


Figure 1.6 Maximum pressure and impulse distribution on the ground for 10 lines (110lb) of Iremite.

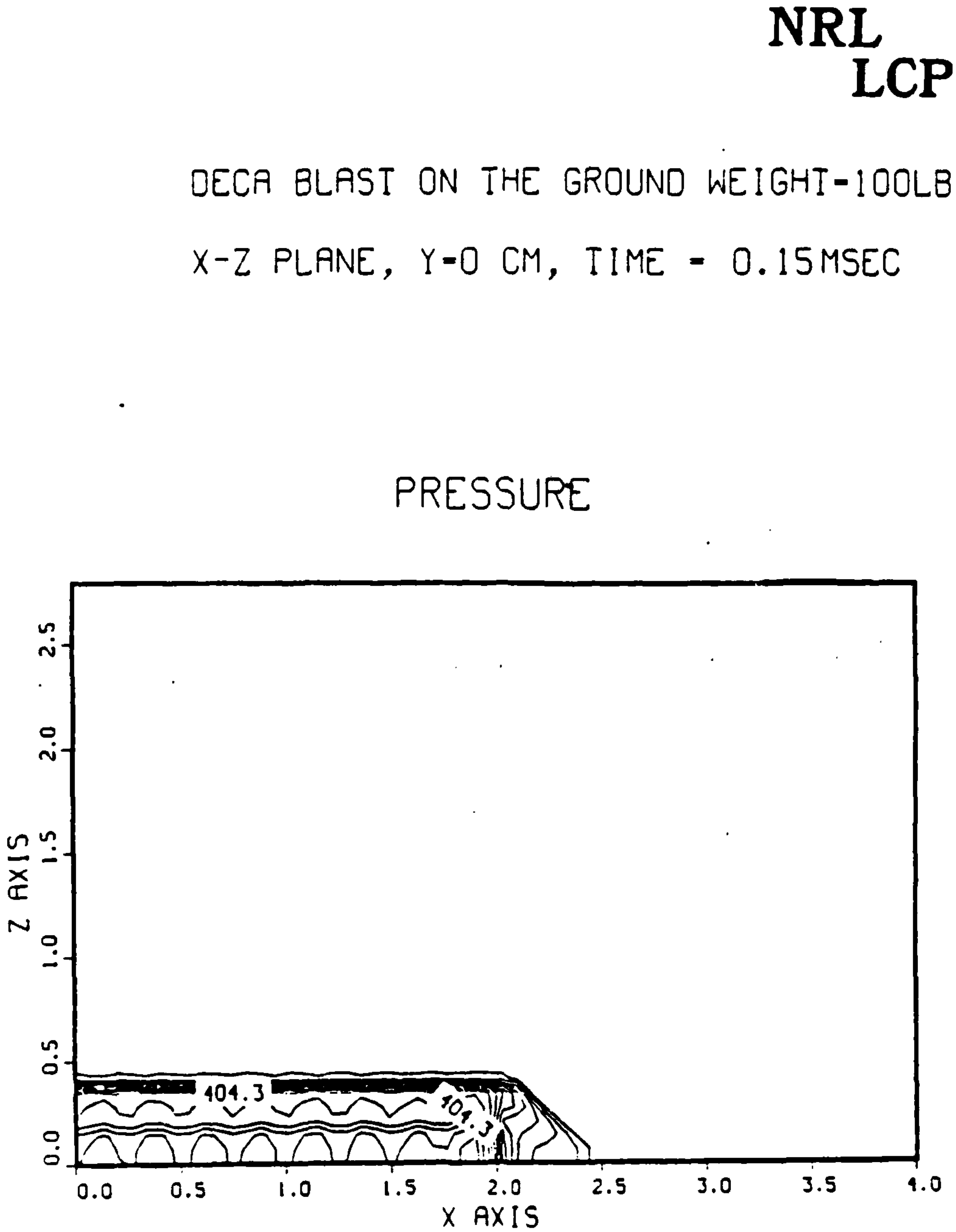


Figure 1.7a Pressure contours for 17 lines $4m \times 4m$ DECA array. X–Z cross section.

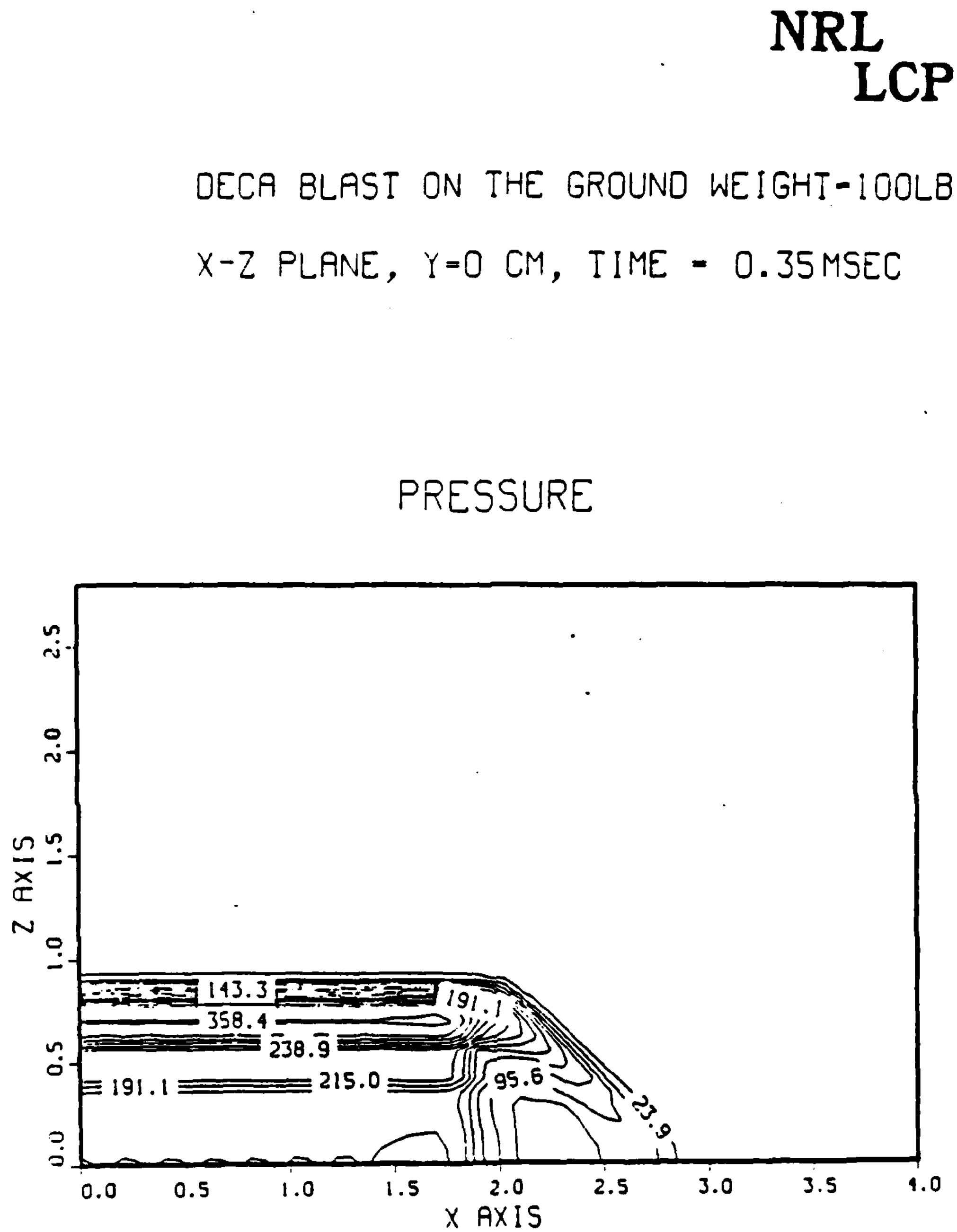


Figure 1.7b Pressure contours for 17 lines $4m \times 4m$ DECA array. X-Z cross section.

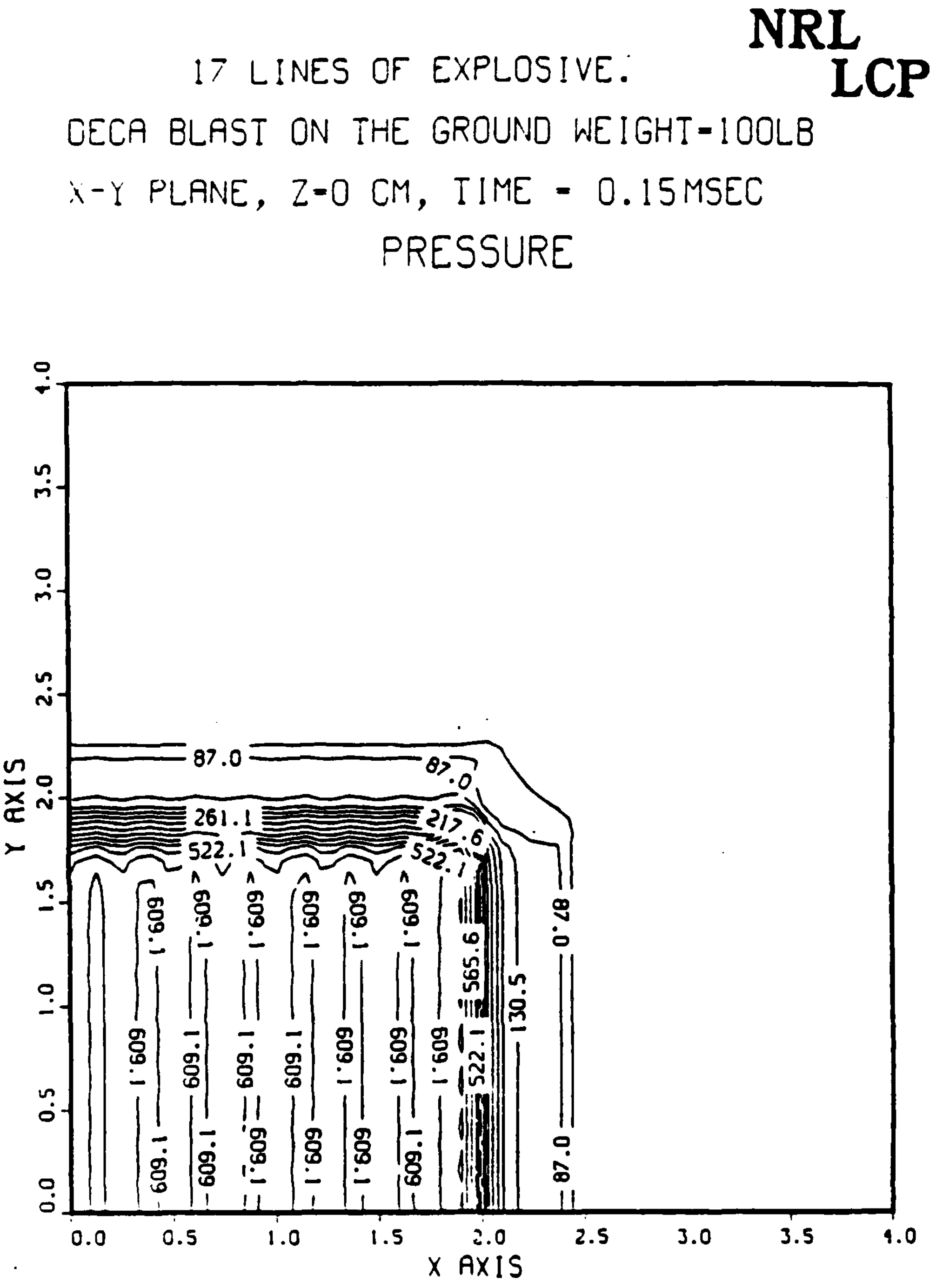


Figure 1.8a Pressure contours for 17 lines $4m \times 4m$ DECA array. X–Y cross section.

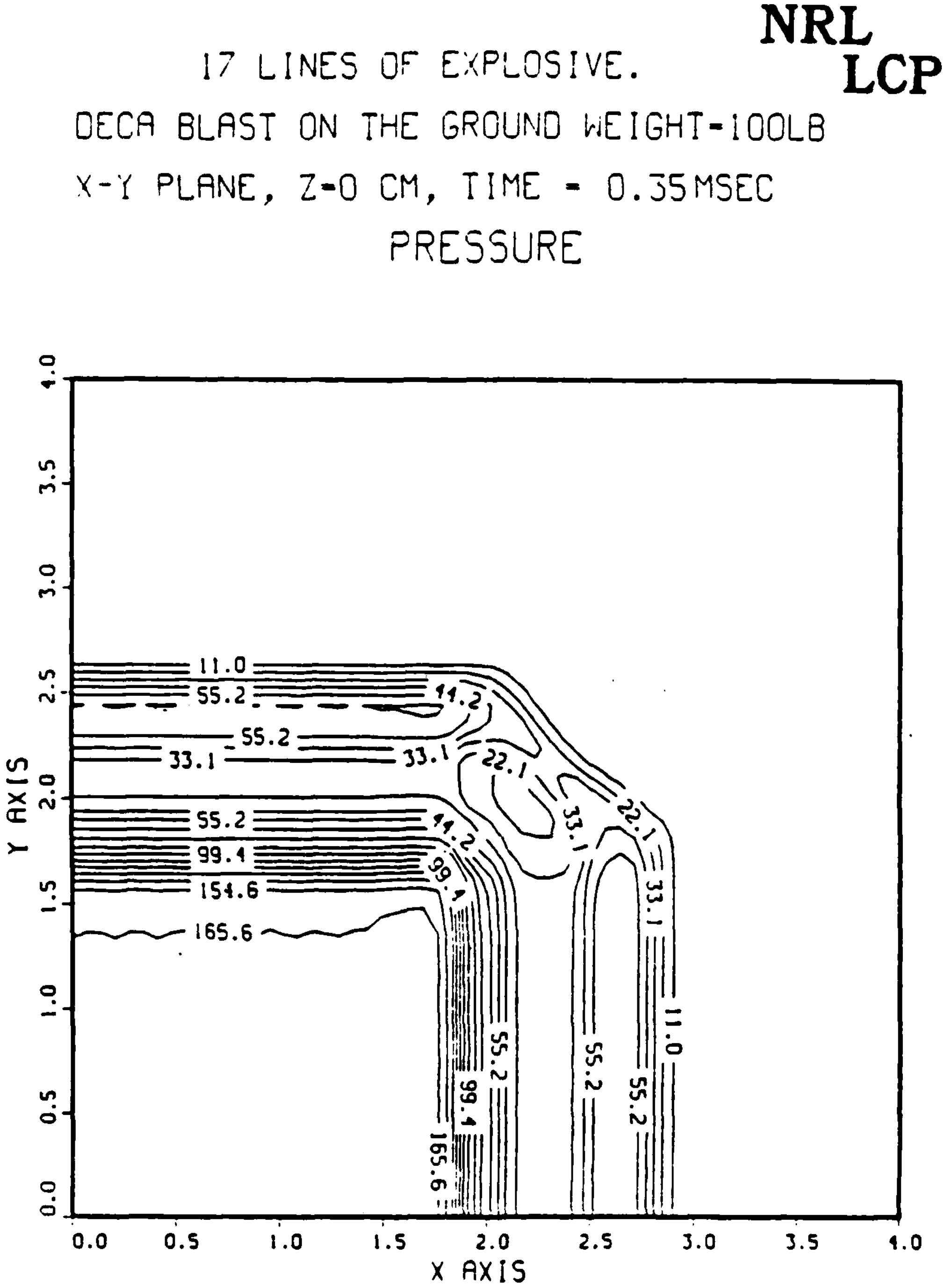


Figure 1.8b Pressure contours for 17 lines $4m \times 4m$ DECA array. X-Y cross section.

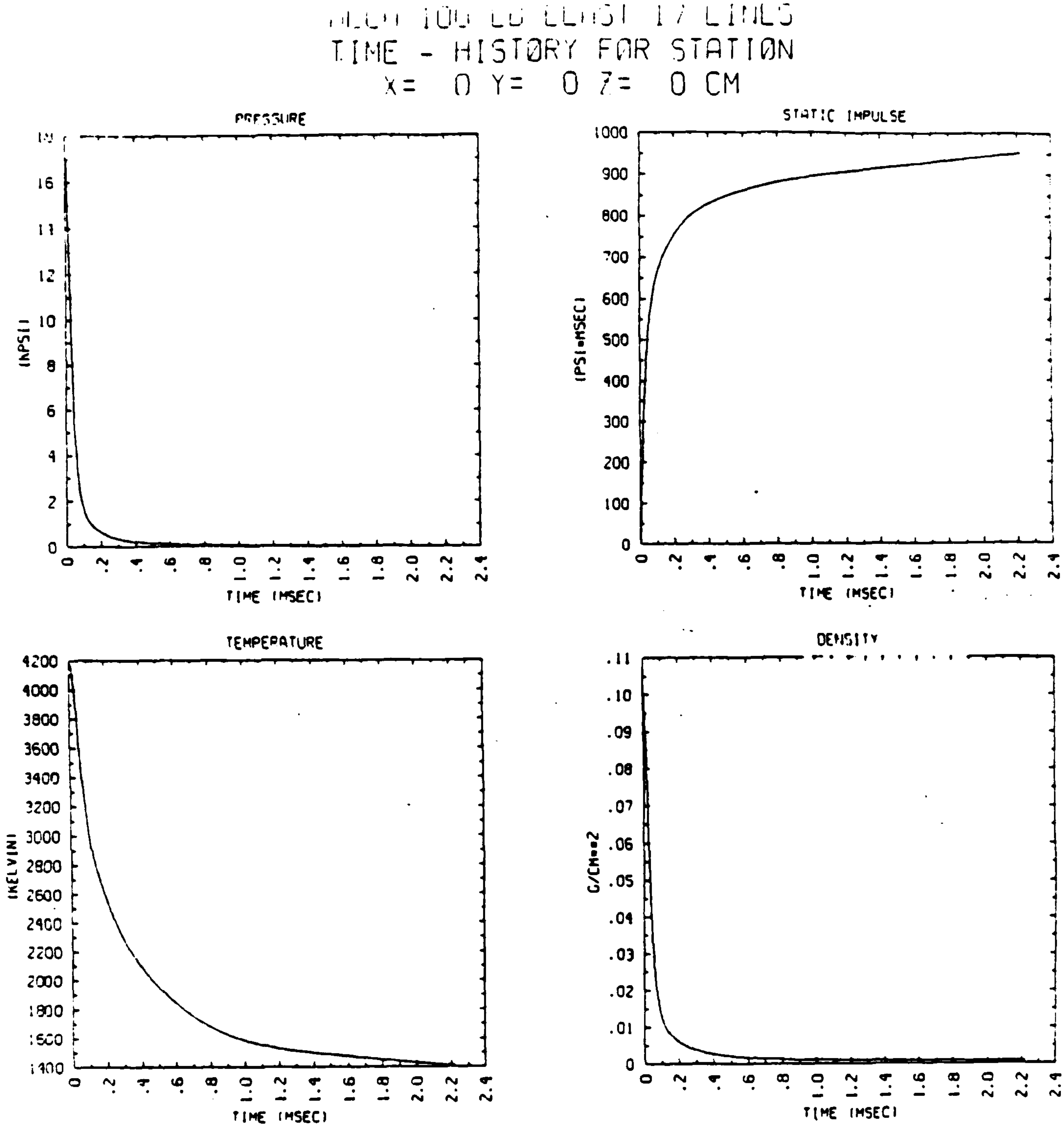


Figure 1.0a Time history of pressure, impulse, temperature and density on the ground under a line of explosive.

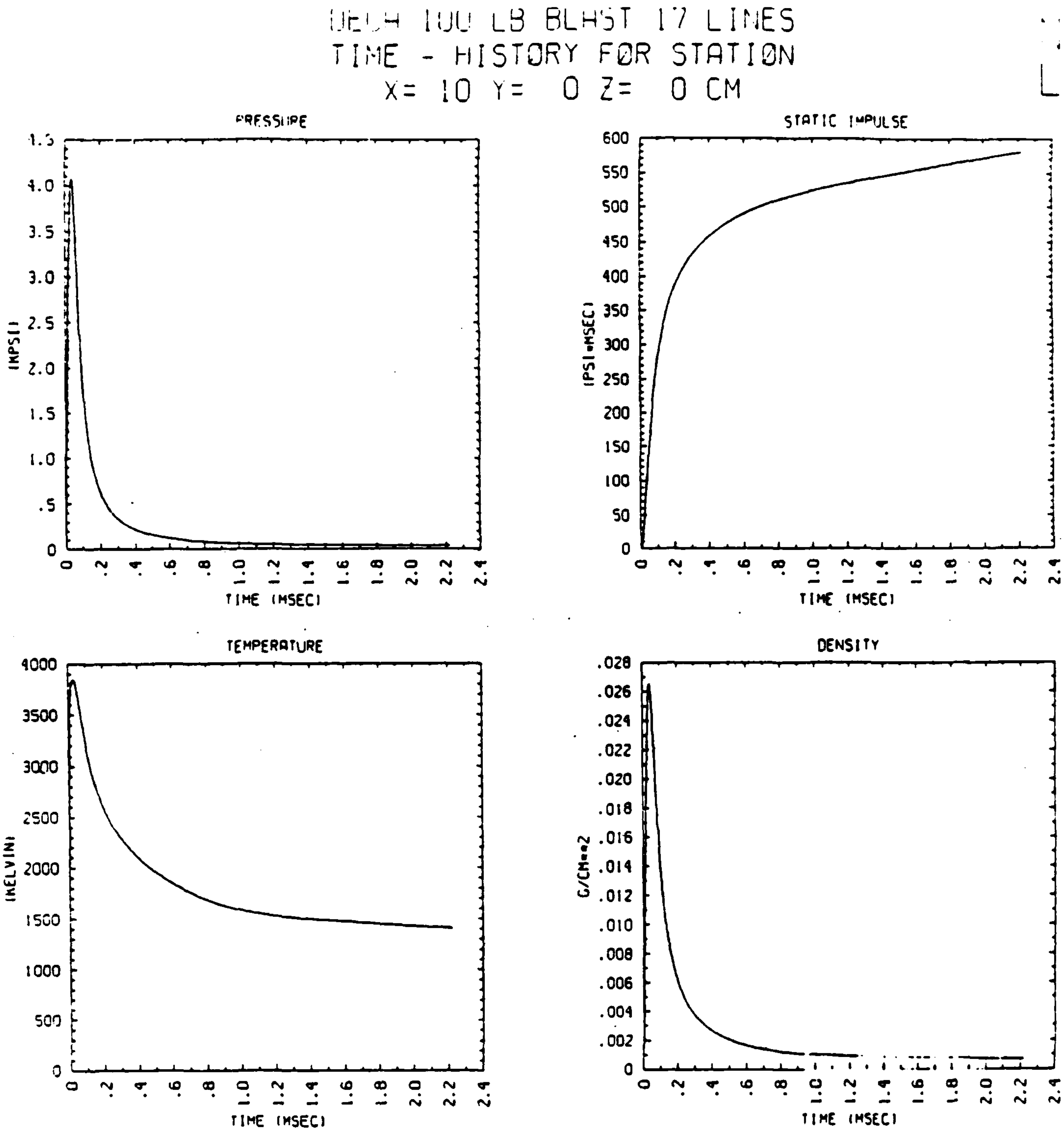


Figure 1.9b Time history of pressure, impulse, temperature and density on the ground between lines of explosive.

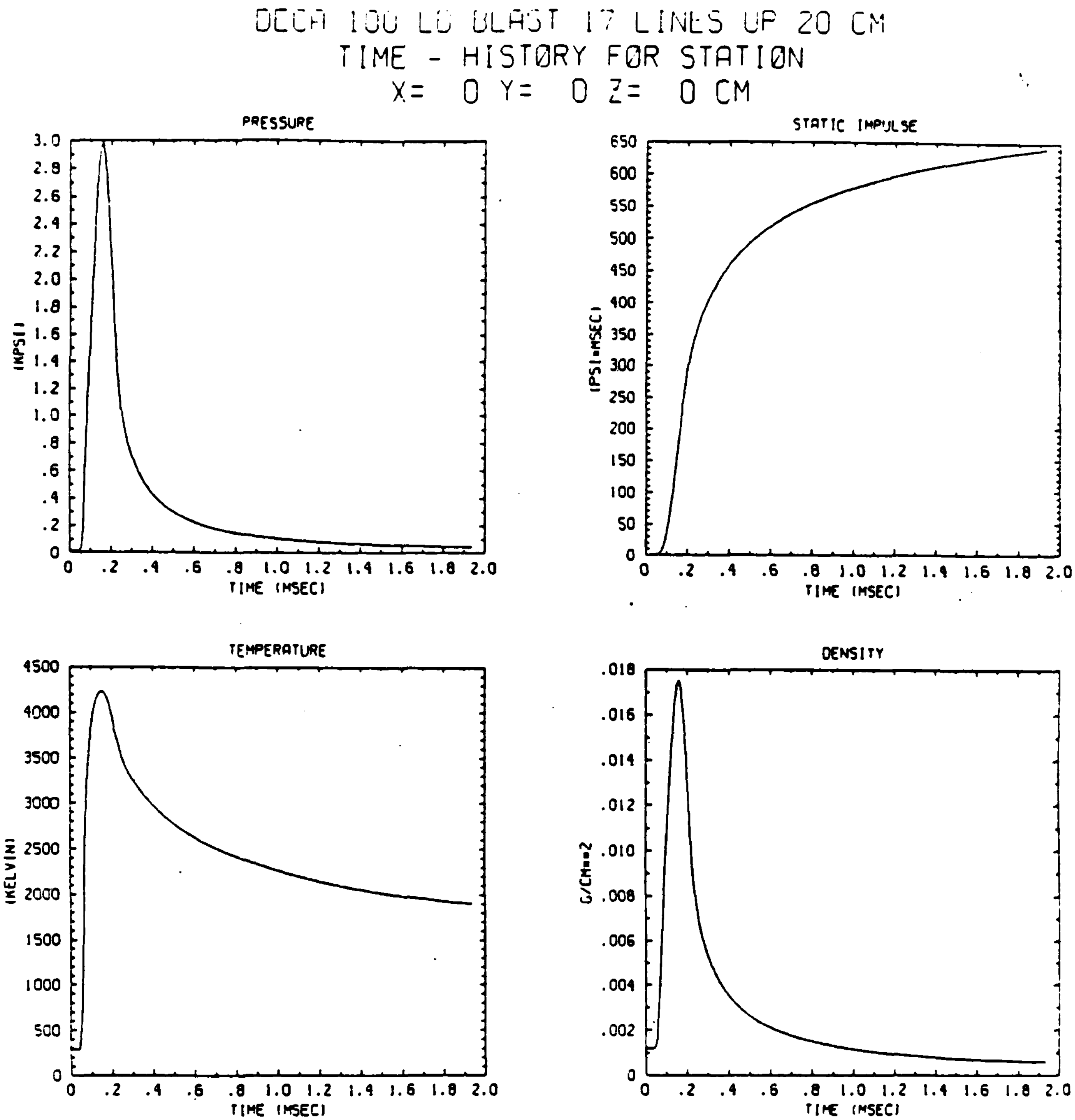


Figure 1.10a Time history of pressure, impulse, temperature and density on the ground for array detonation 20cm above the ground.

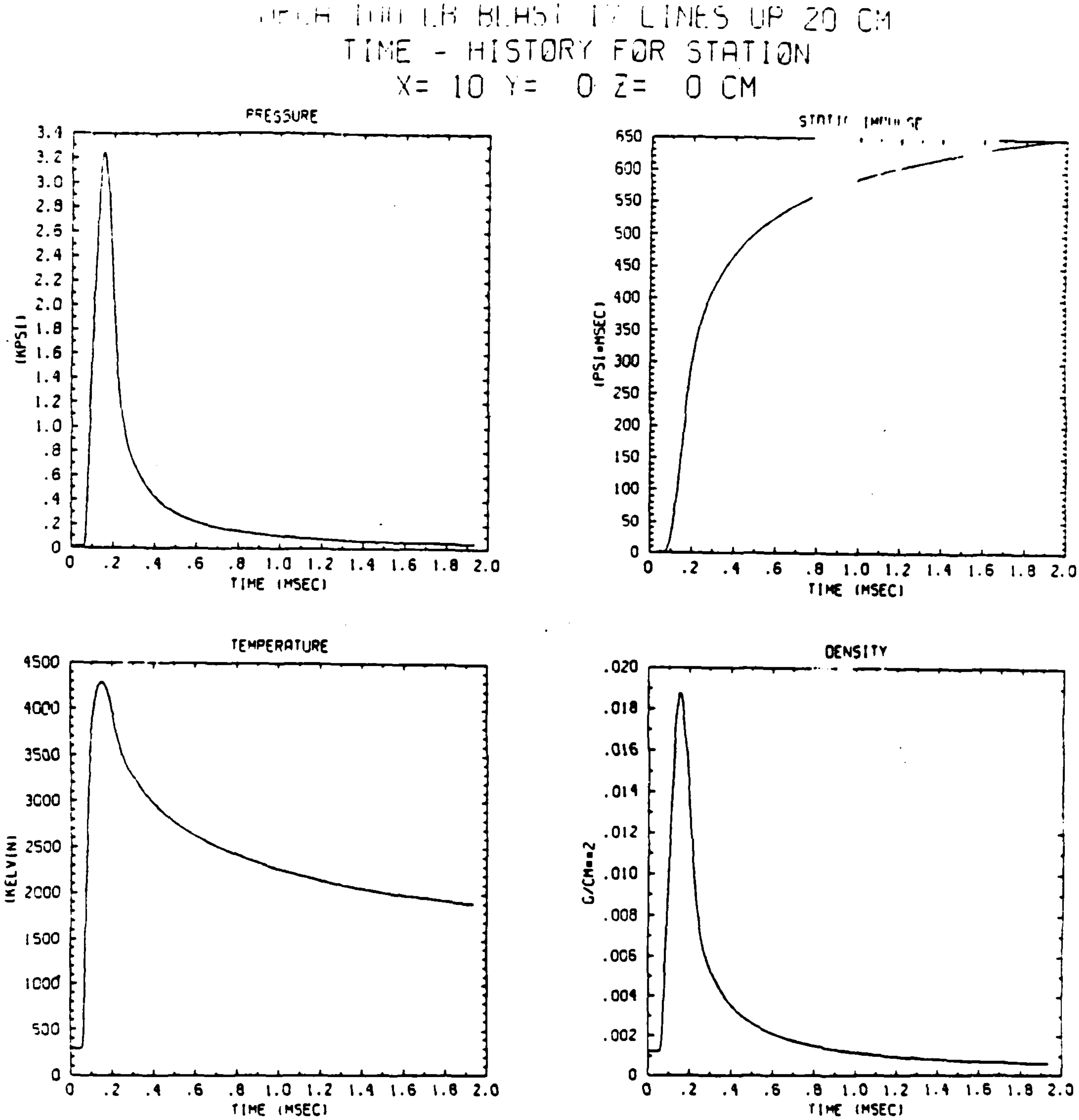


Figure 1.10b Time history of pressure, impulse, temperature and density on the ground for array detonation 20cm above the ground.

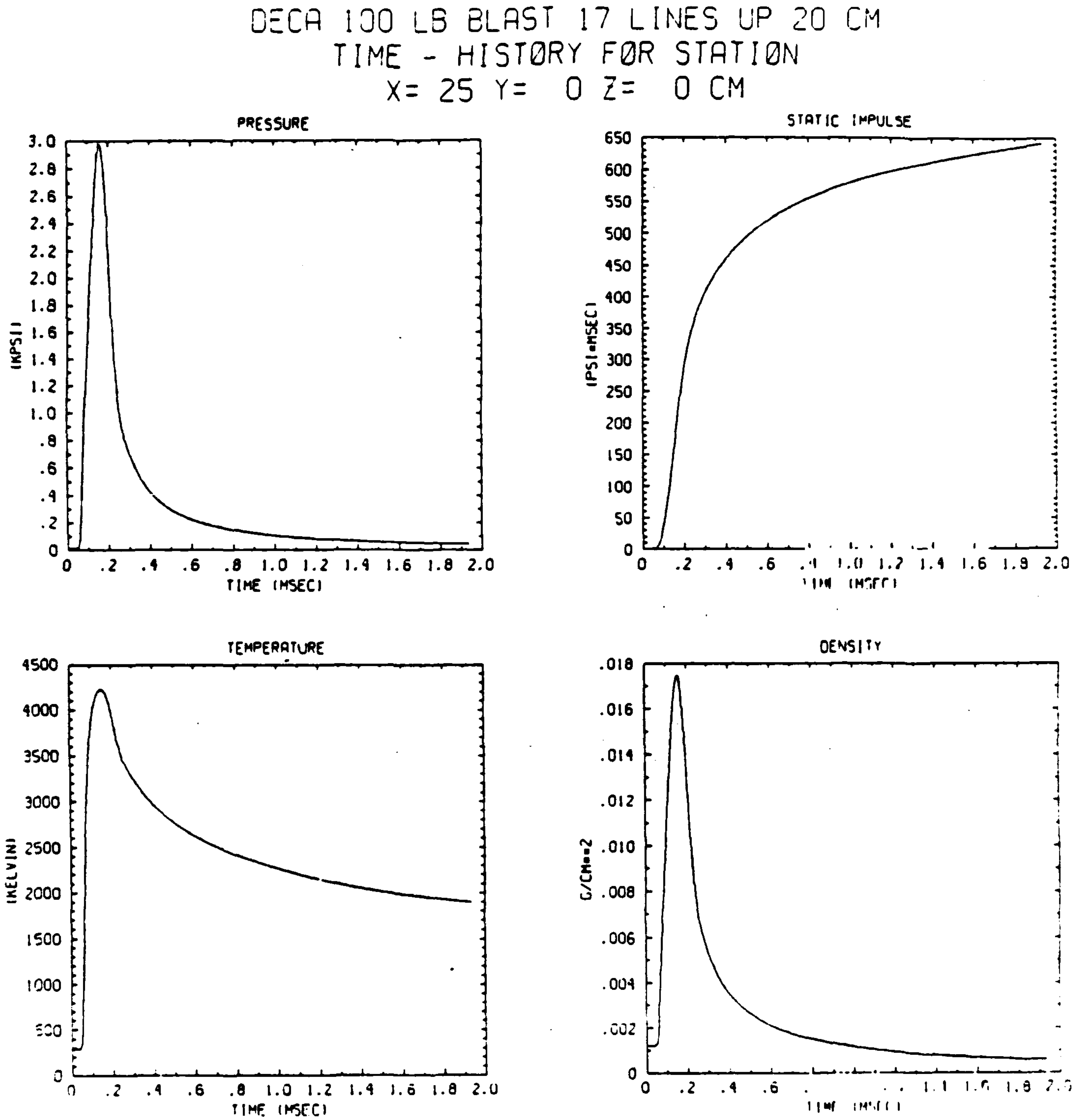


Figure 1.10c Time history of pressure, impulse, temperature and density on the ground for array detonation 20cm above the ground.

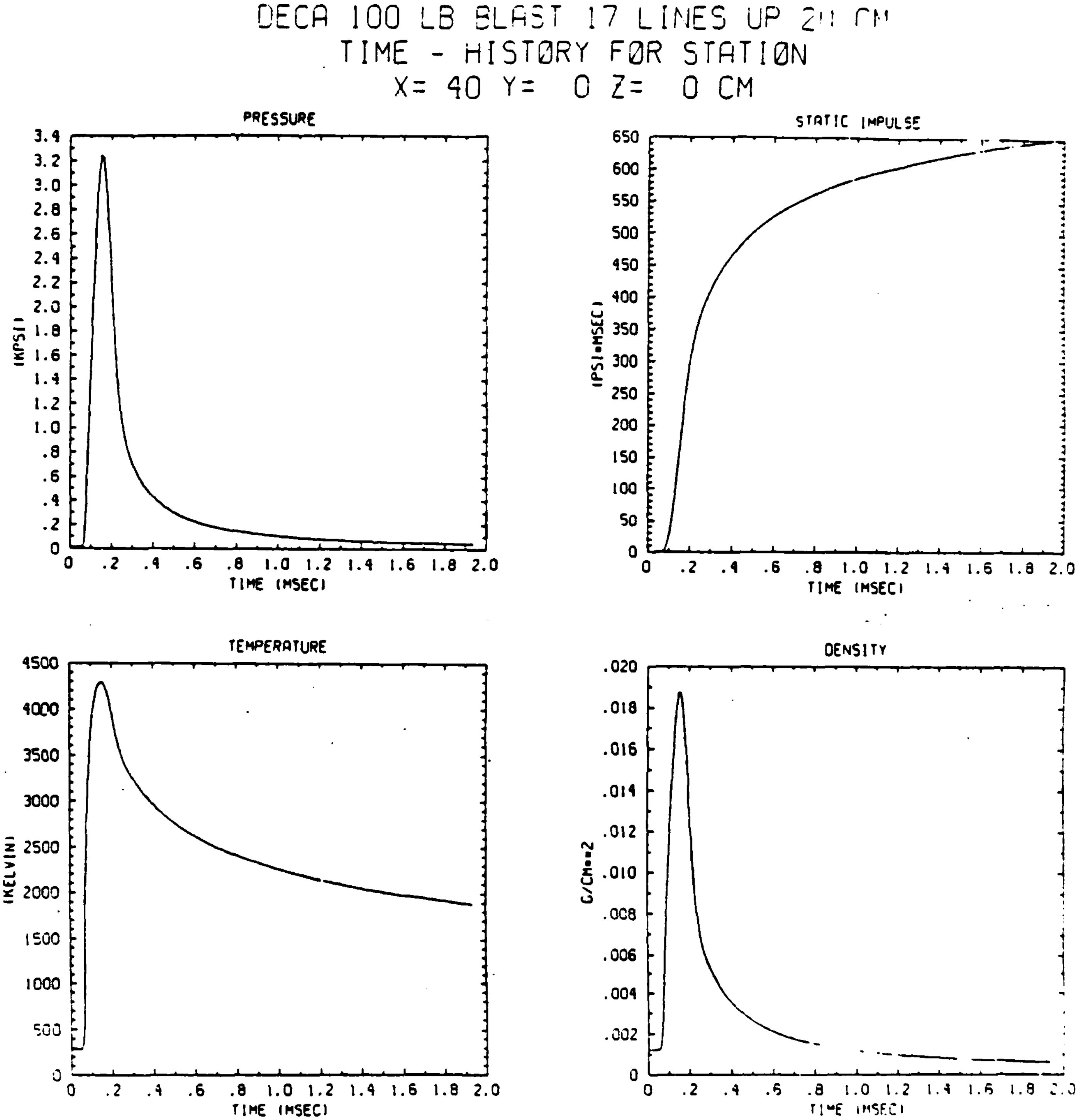


Figure 1.10d Time history of pressure, impulse, temperature and density on the ground for array detonation 20cm above the ground.

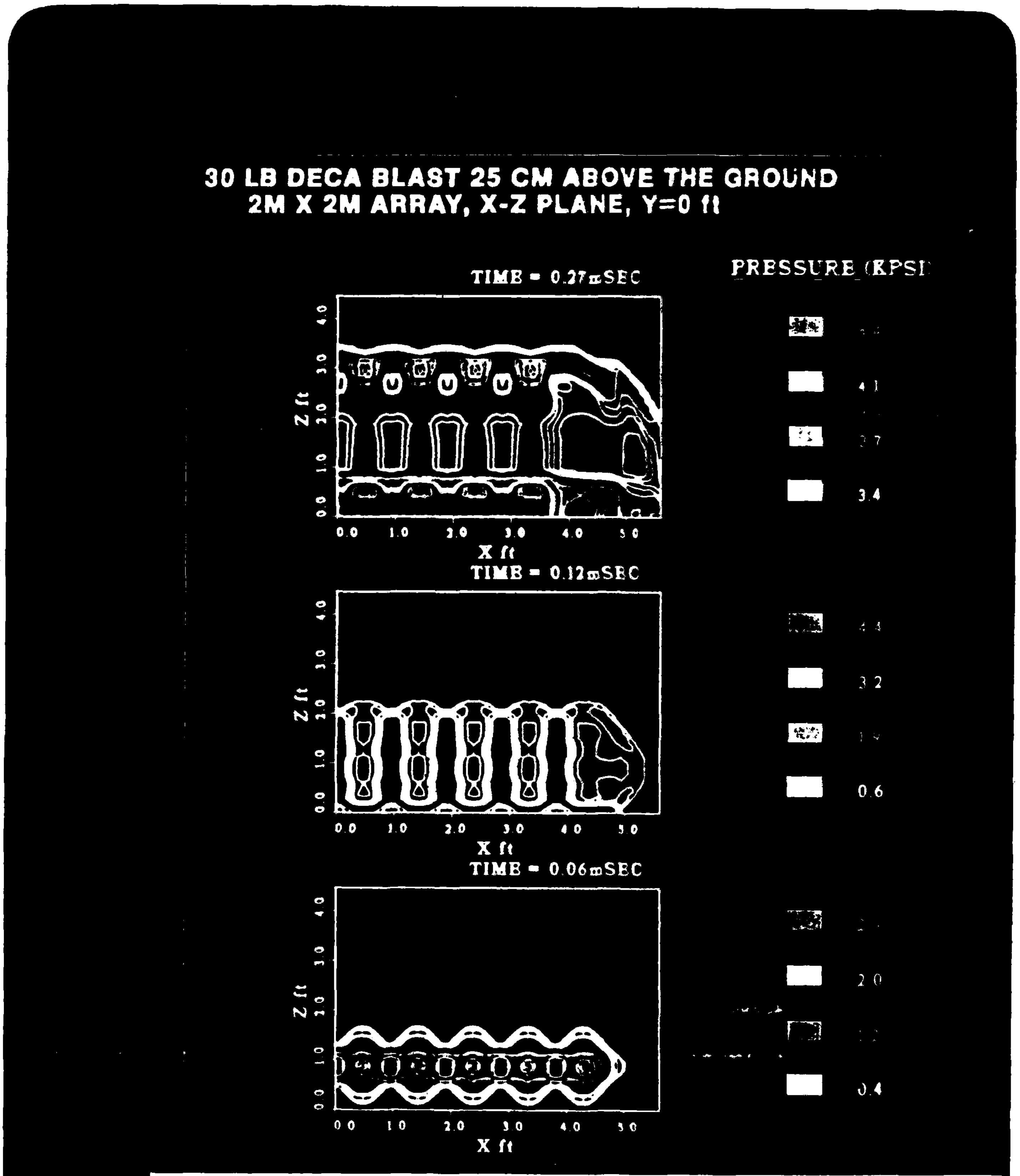


Figure 1.11 Pressure contours for 10 lines (2.25m × 2.0m) Primacord detonation.

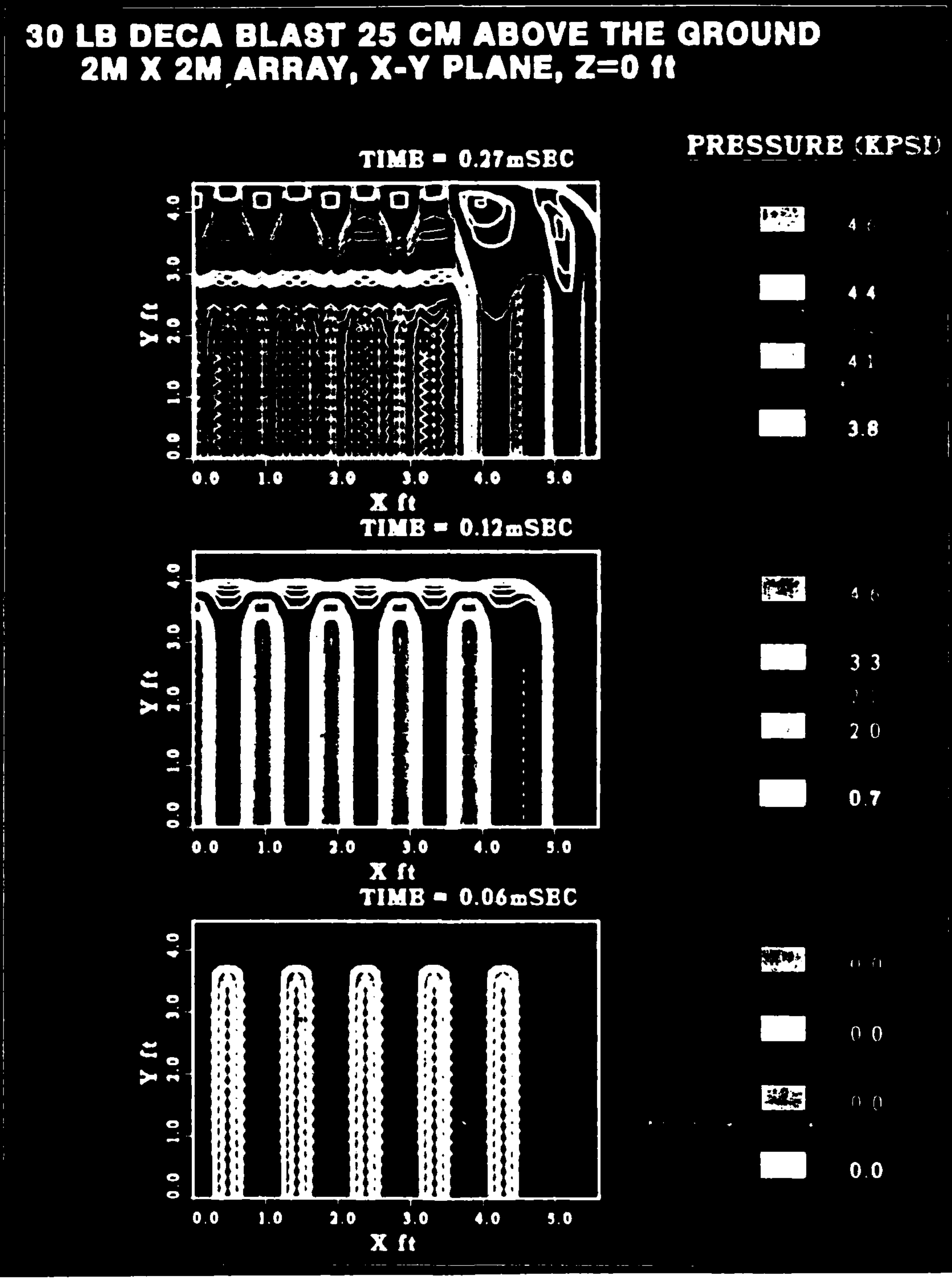


Figure 1.12 Pressure contours for 10 lines (2.25m × 2.0m) Primacord detonation.

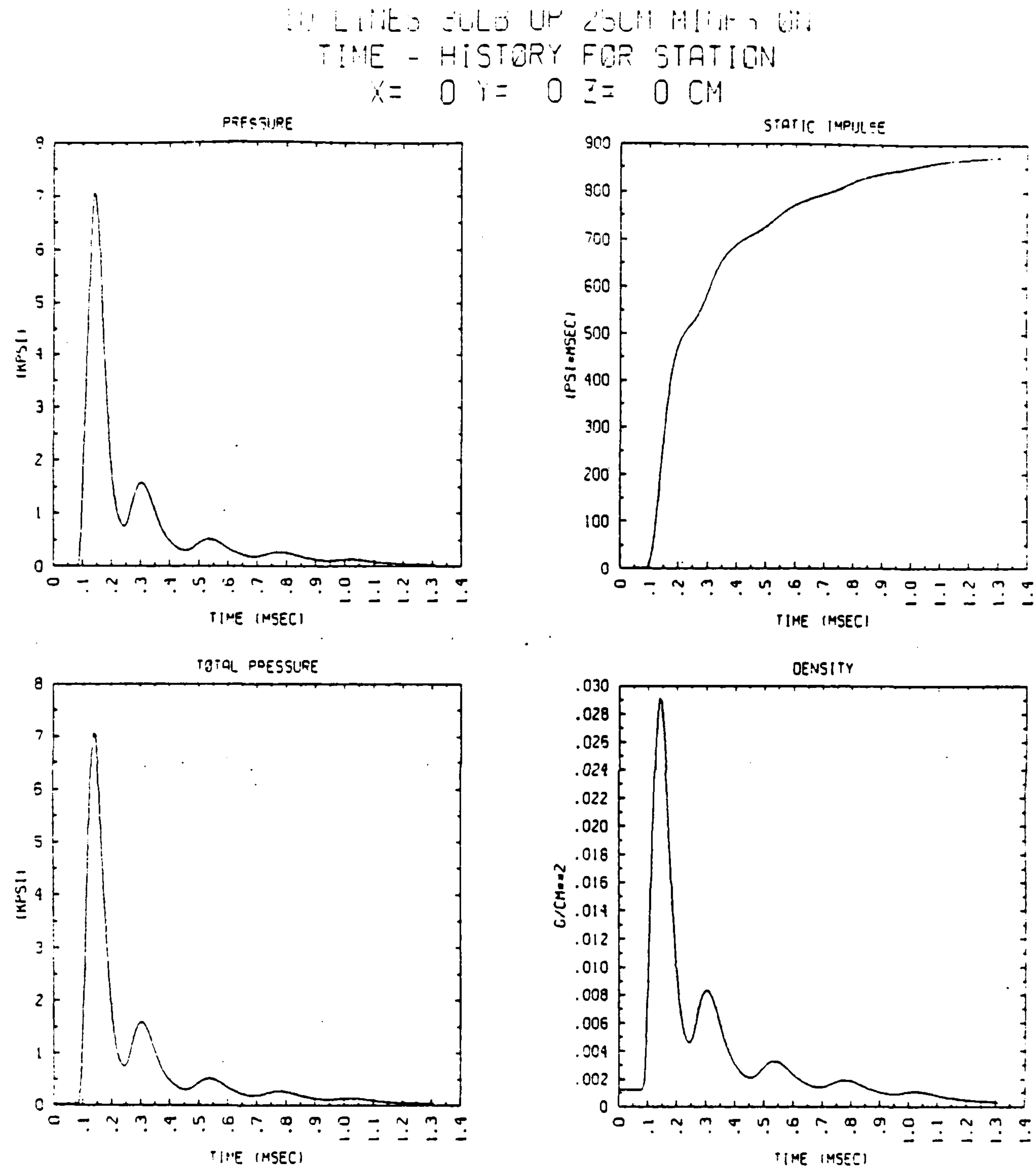


Figure 1.13a Time history for pressure, impulse, total pressure and density on the ground. X=0, Y=0, Z=0

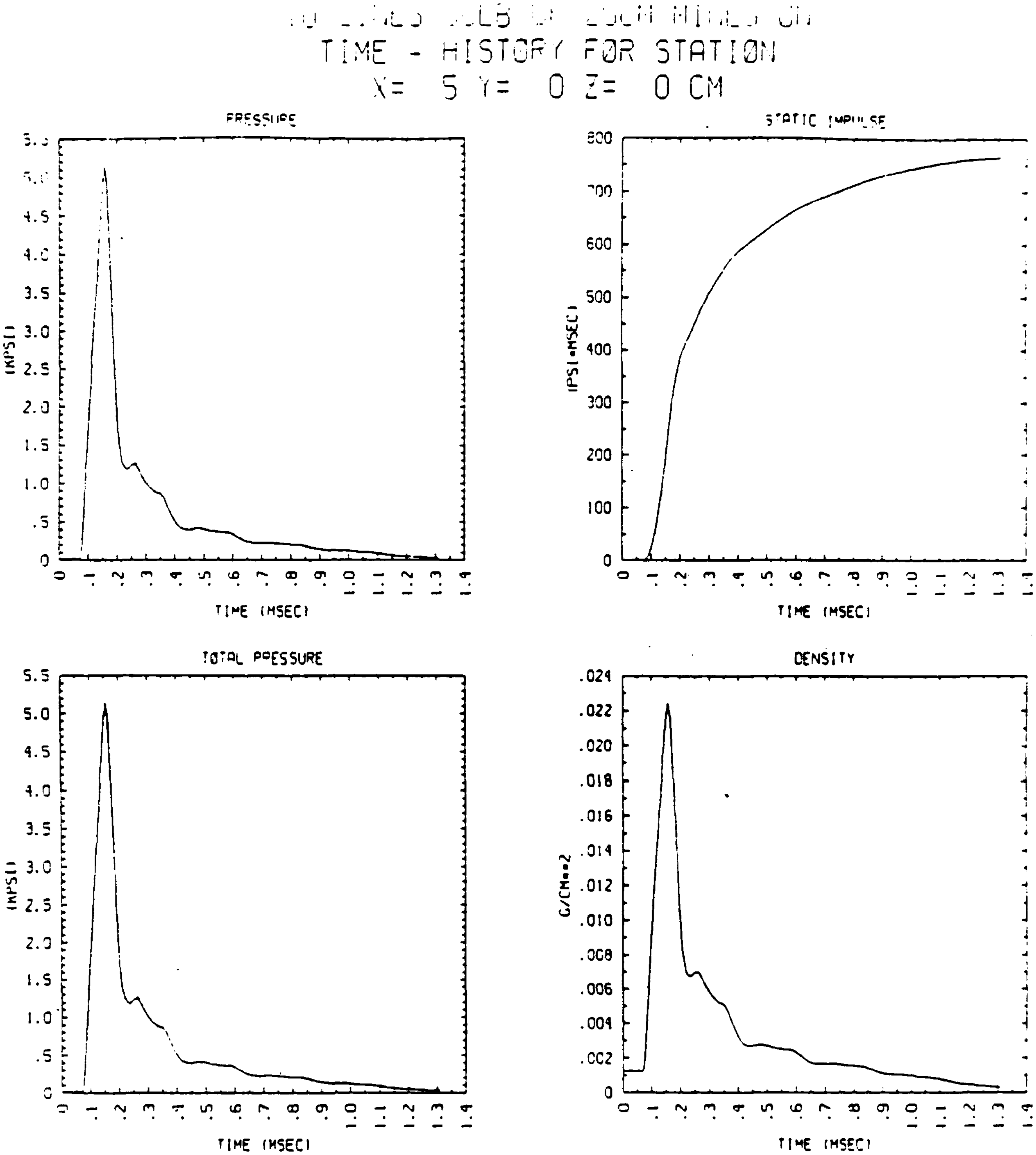


Figure 1.13b Time history for pressure, impulse, total pressure and density on the ground. X=5, Y=0, Z=0

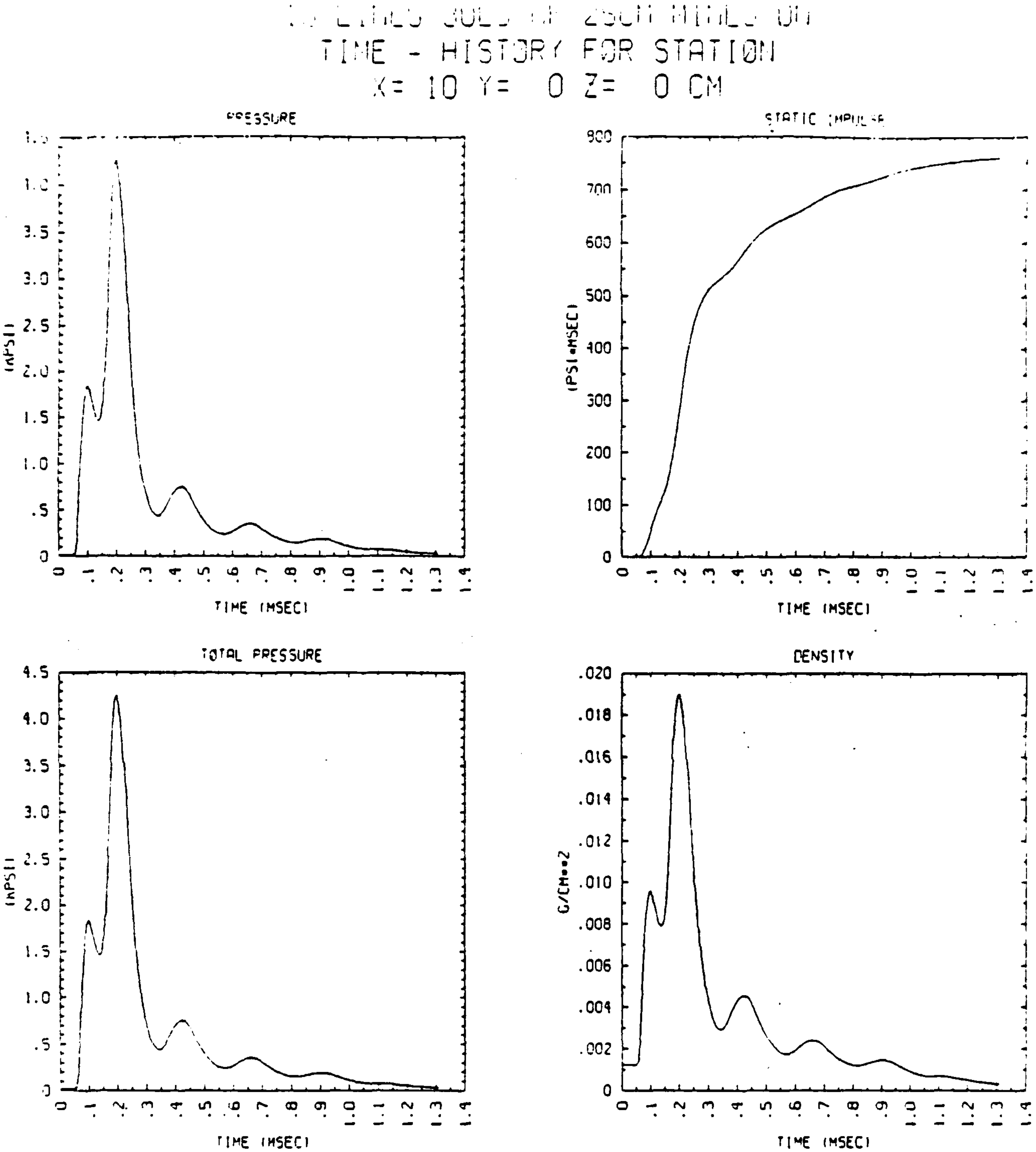


Figure 1.13c Time history for pressure, impulse, total pressure and density on the ground. X=10, Y=0, Z=0

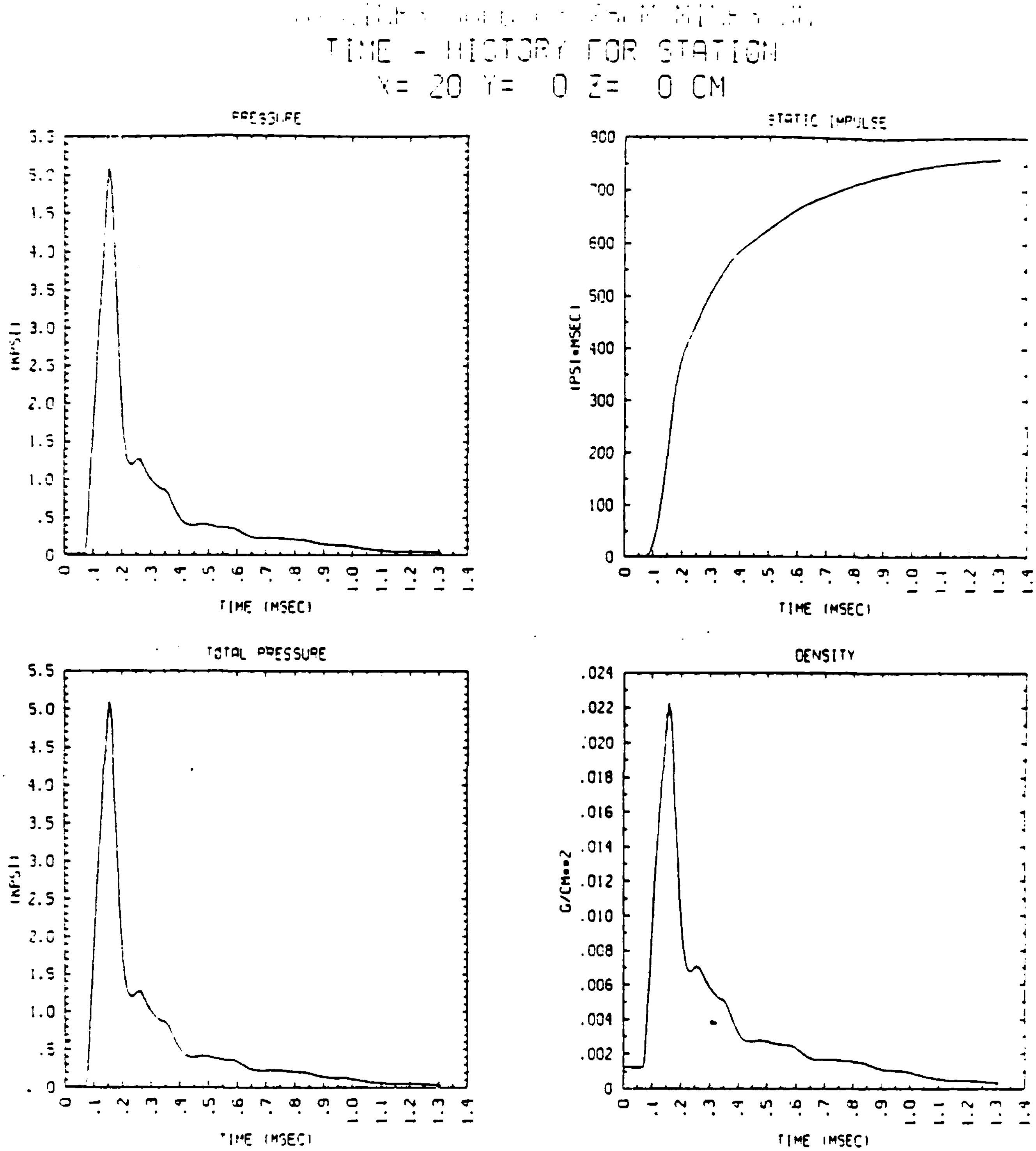


Figure 1.13d Time history for pressure, impulse, total pressure and density on the ground. X=20, Y=0, Z=0

COMPARISON OF M58, FAE AND DECA

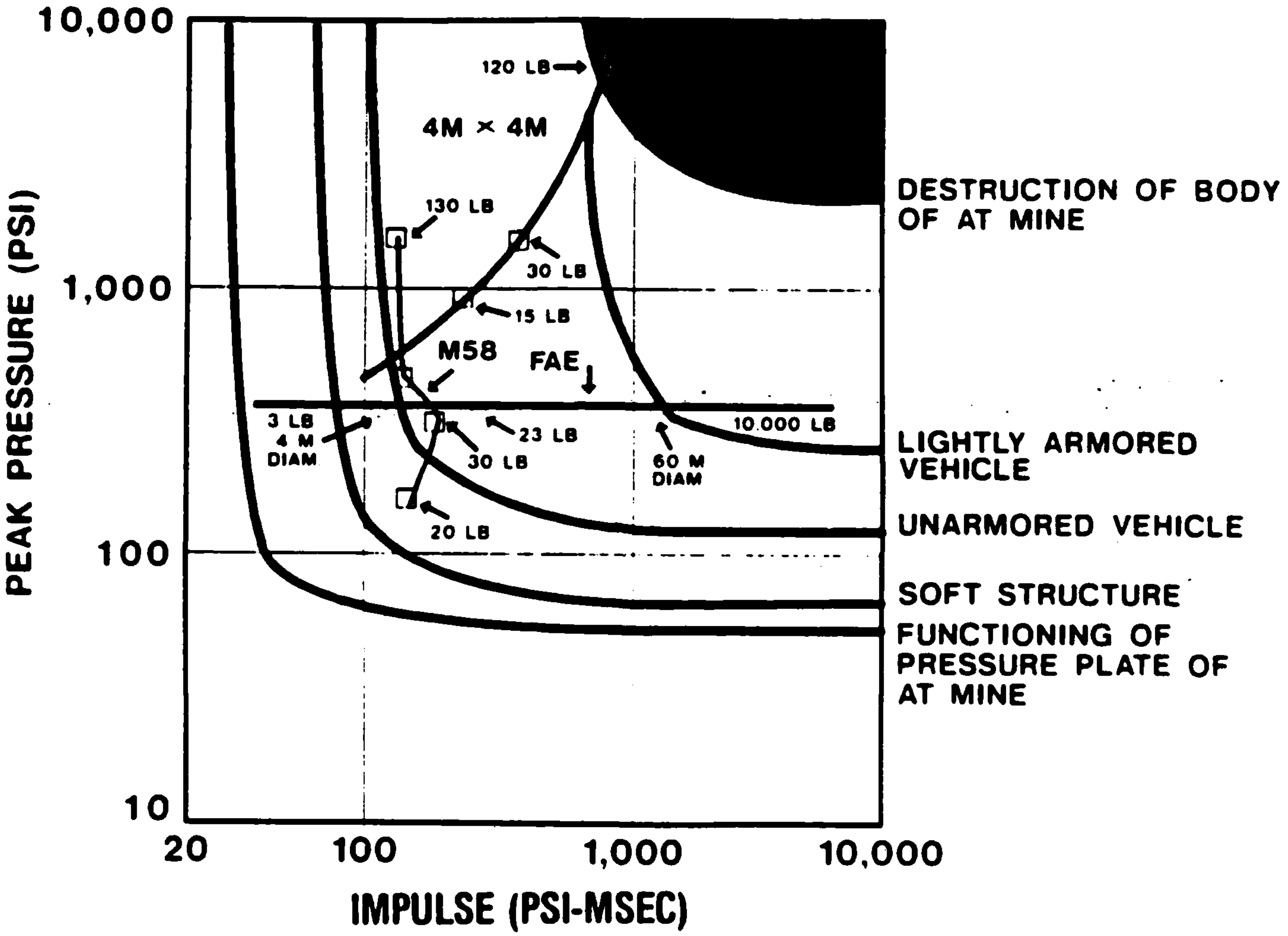


Figure 1.14 Maximum pressure and impulse as function of surface (volume for FAE) density of explosive.

Table 1. SUMMARY OF TEST RESULTS

Test No.	Explosive	No. of Lines	Spacing (m)	Height Above Ground (m)	Area: Width Length (m x m)	Areal Density (kg/m^2)	Peak Pressure (kPa) 1 m Above Ground	Peak Pressure (kPa) 2 m Above Ground	Condition of Inert Mine
1	Iremite	6	0.5	0	2.5 x 4.5	2.6	-[a]	-[a]	
2	Iremite	10	0.5	0	4.5 x 4.5	2.6	-[a]	-[a]	
3	Iremite	10	0.5	0	4.5 x 4.5	2.6	-[a]	-[a]	
4	Iremite	10	0.5	0	4.5 x 4.5	2.6	445	2517	
5	Iremite	10	0.5	0	4.5 x 4.5	2.6	425	2172	
6	Iremite	10	0.5	0	4.5 x 4.5	3.9	3964	2241	
7	Iremite	10	0.5	0.25	4.5 x 4.5	2.6	2861	1517	
8	Primacord	10	0.25	0.25	2.25 x 2.5	2.4	-[b]	-[b]	Lightly damaged
9	Primacord	10	0.25	0.25	2.25 x 2.5	2.4	4826	2054	Pressure plate destroyed, mine case crushed
10	Primacord	12	0.25	0.50	2.75 x 2.5	2.4	5722	2606	Pressure plate severely damaged, mine slightly crushed

[a] No useful data were obtained because of severe vibrations in instrumentation poles.

[b] Explosive did not fully detonate (about 50% detonation).

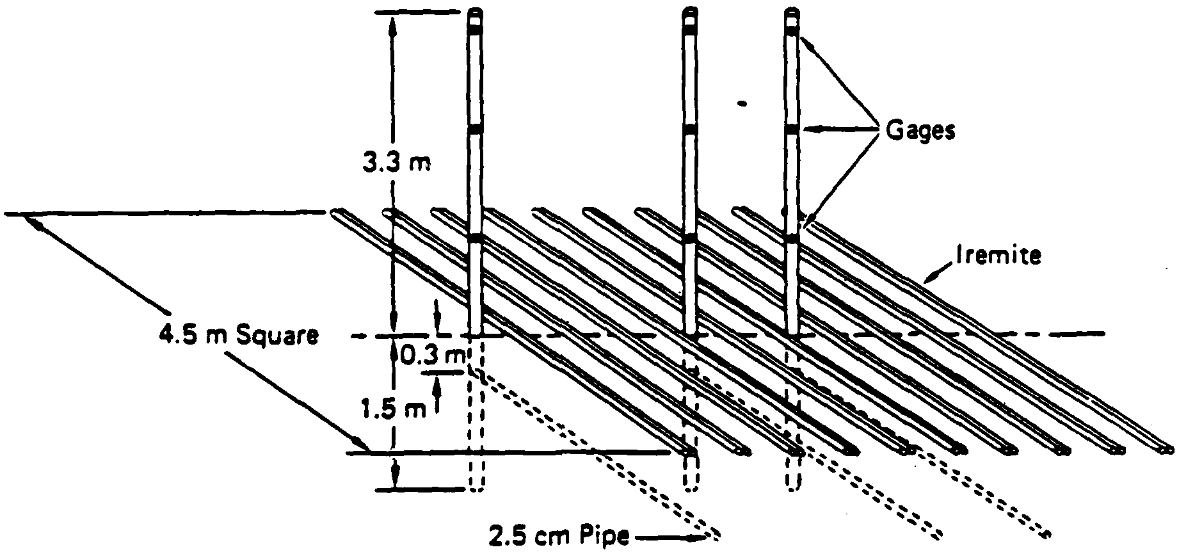


Figure 2.1 Prespective of experimental configuration.

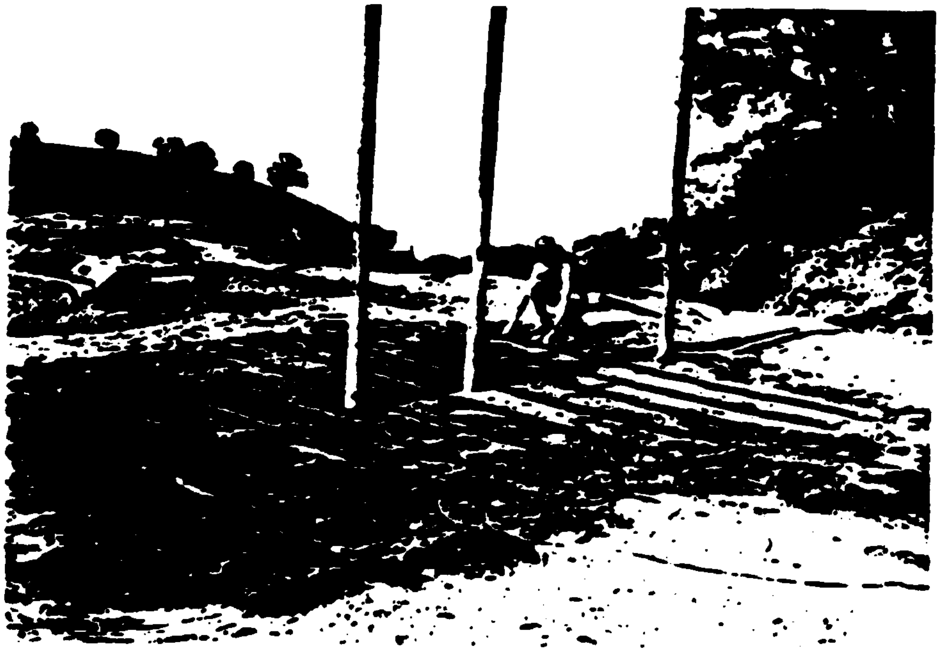


Figure 2.2 Explosive array and insrumentation poles.

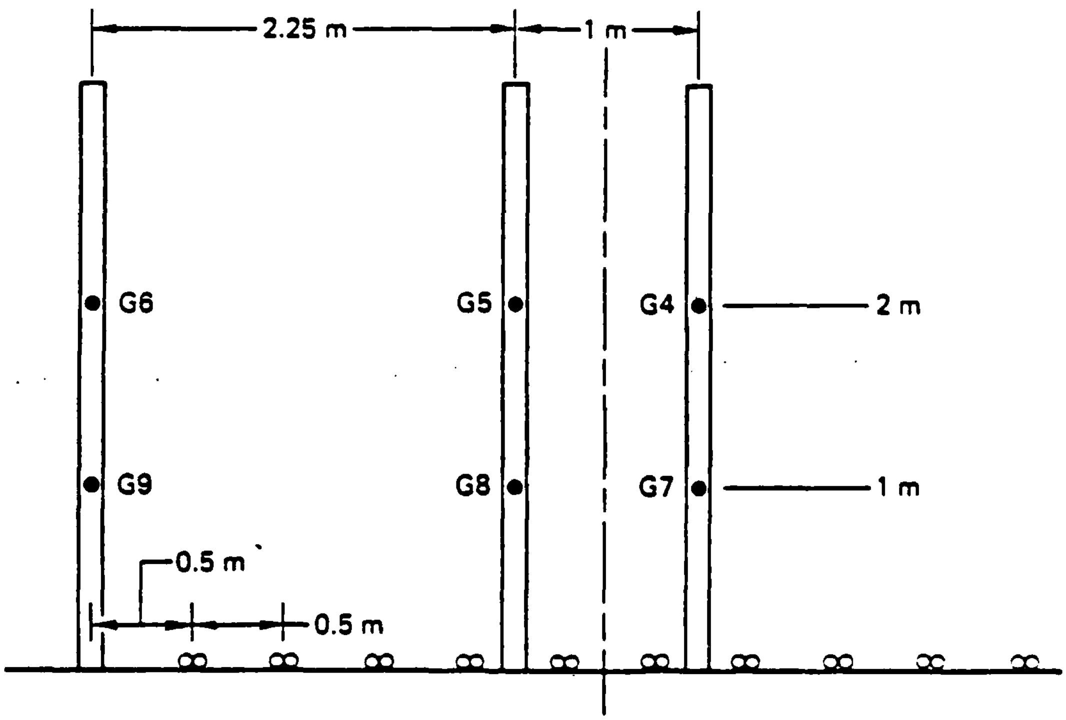
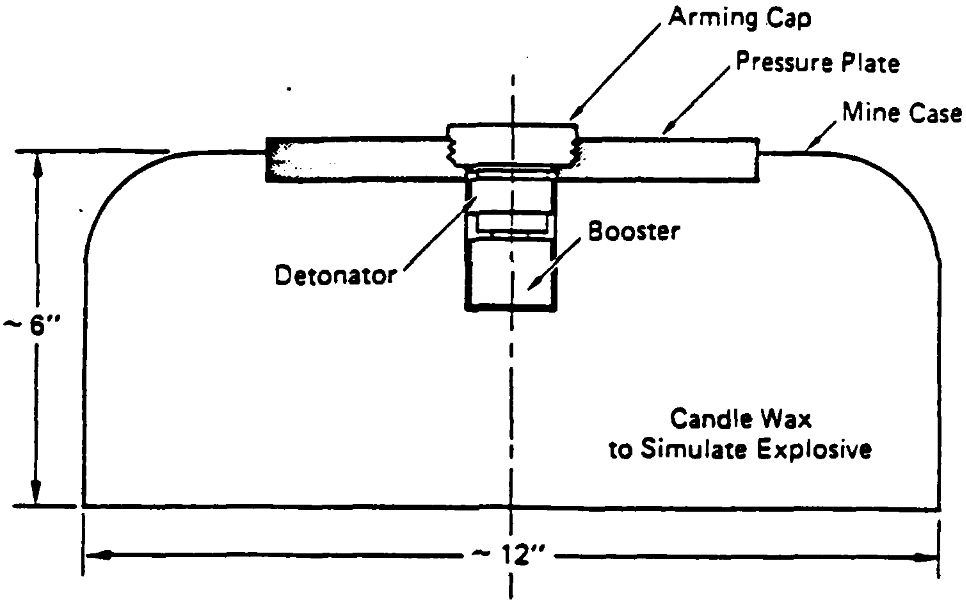
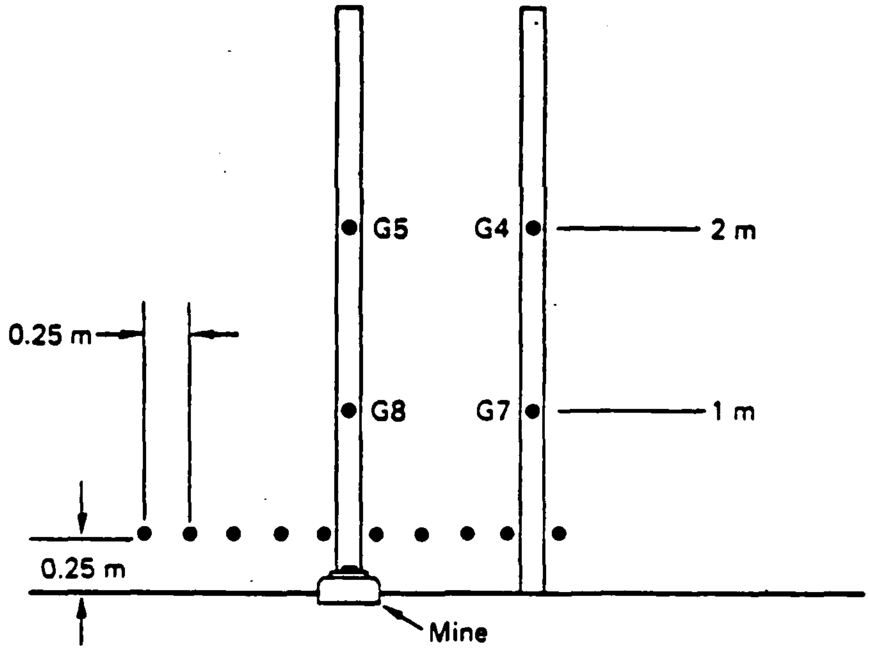


Figure 2.3 Experimental setup showing instrumentation poles.

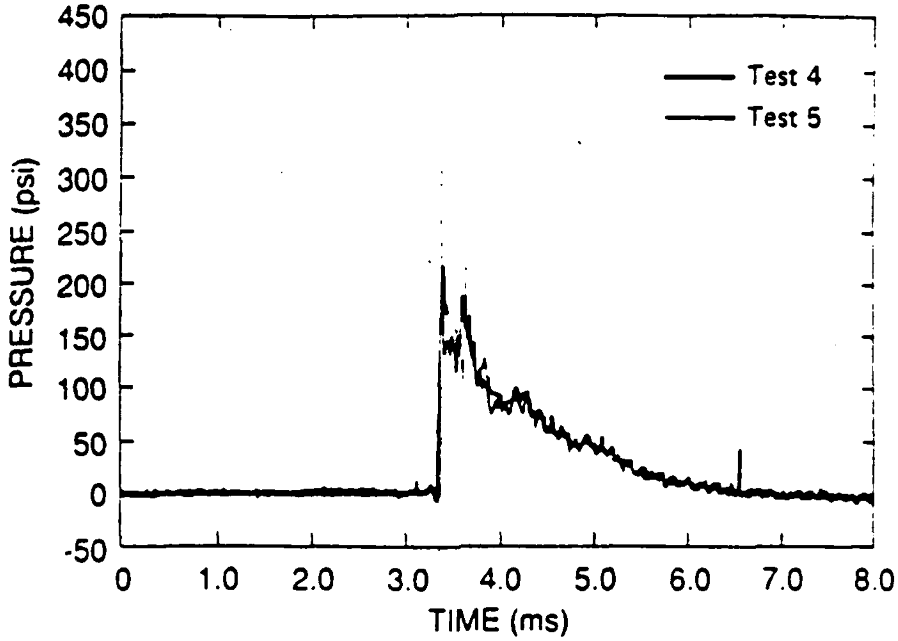


(a) Cross Section of Antitank Mine

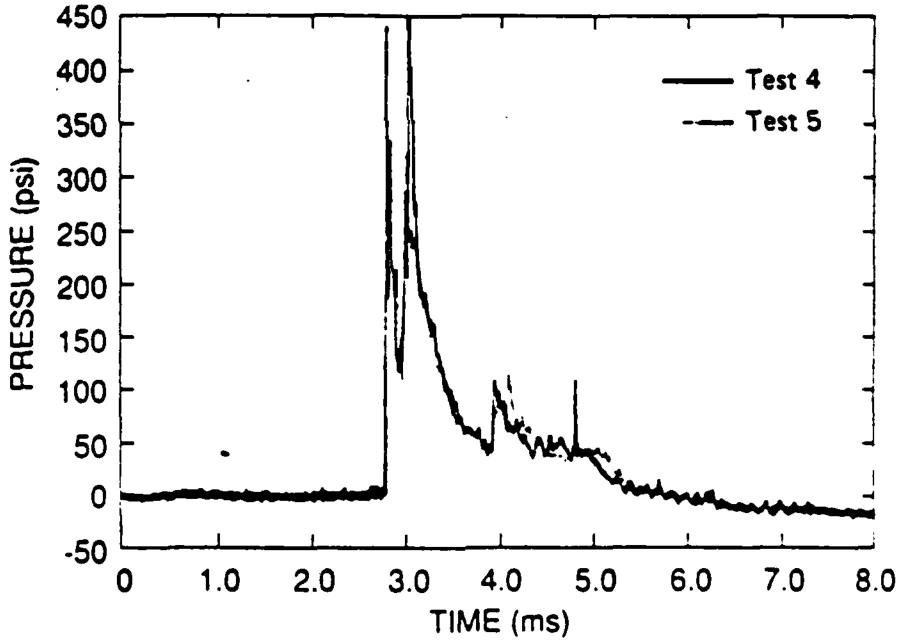


(b) Explosive Array and Instrumentation

Figure 2.4 Experimental configuration for antitank mine tests.



(a) Gage G4 (2 m above soil)



(b) Gage G7 (1 m above soil)

Figure 2.5 Comparison of tests 4 and 5.

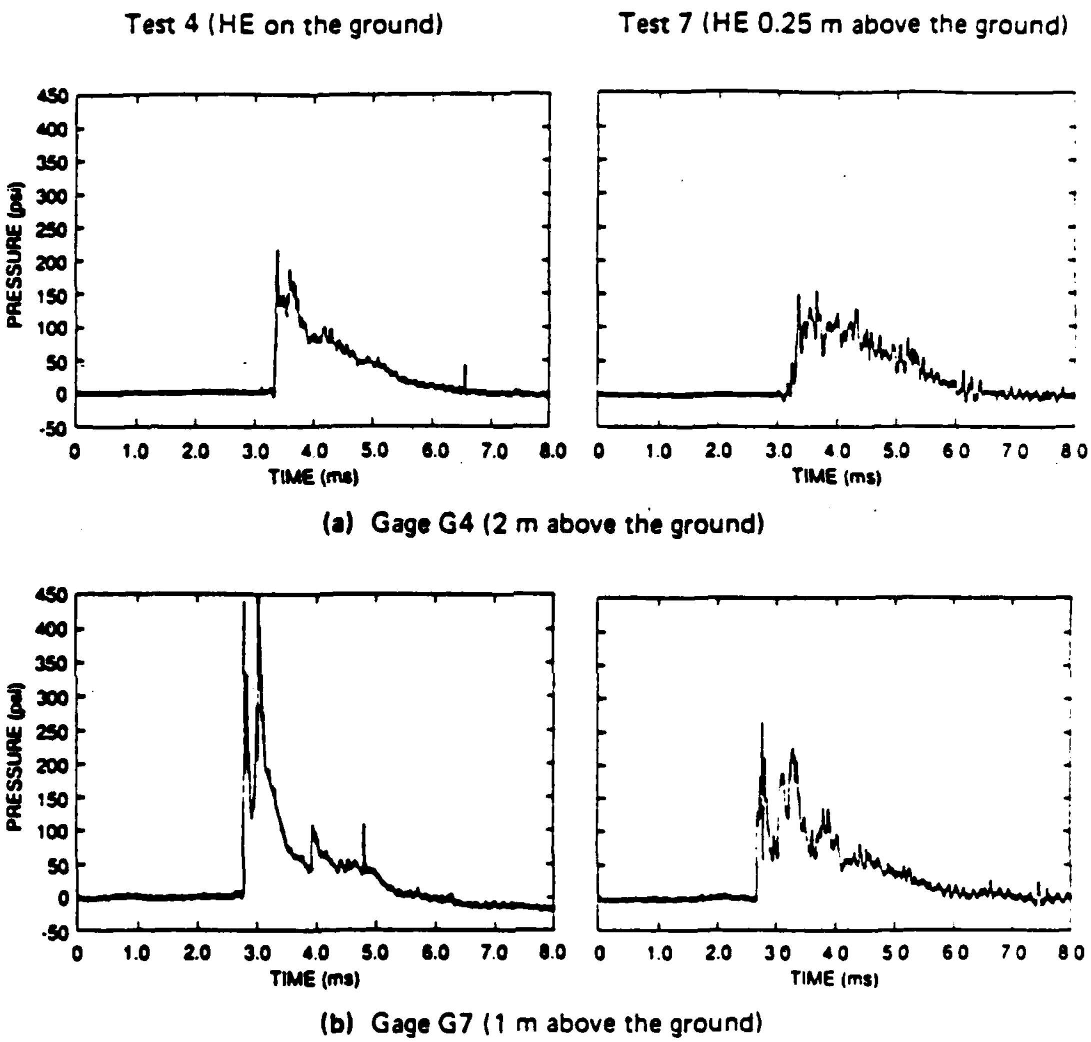
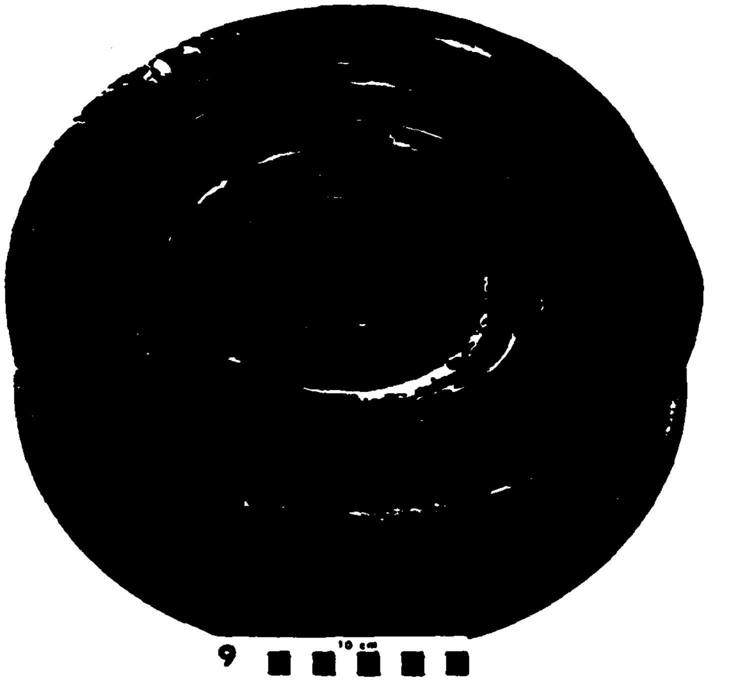
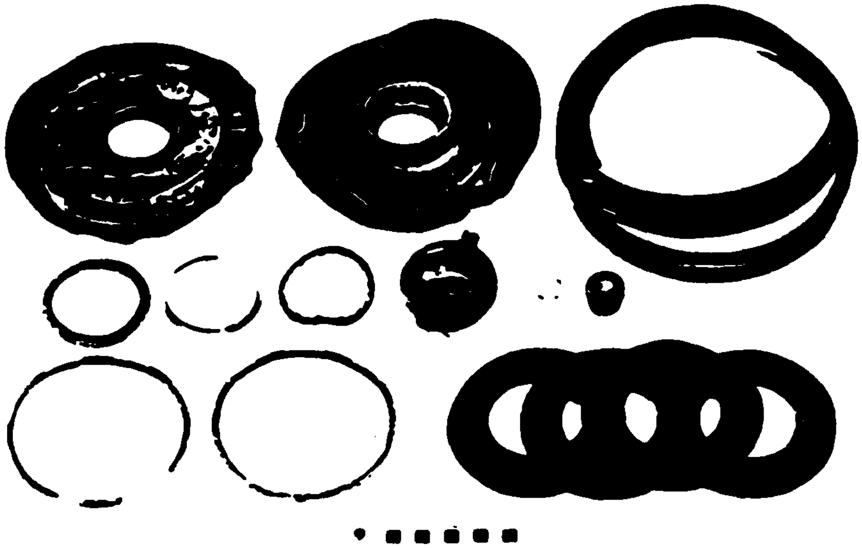


Figure 2.6 Comparison of data to show effects of suspending explosive above the ground.

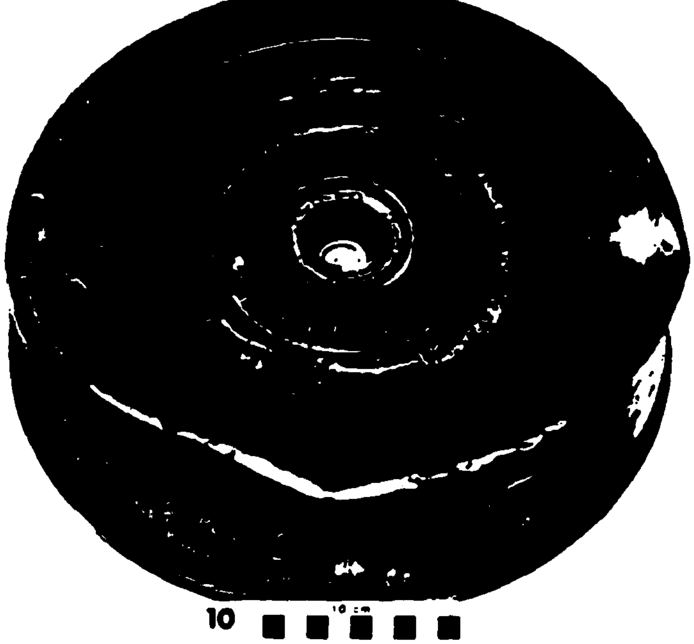


(a) Mine Casing

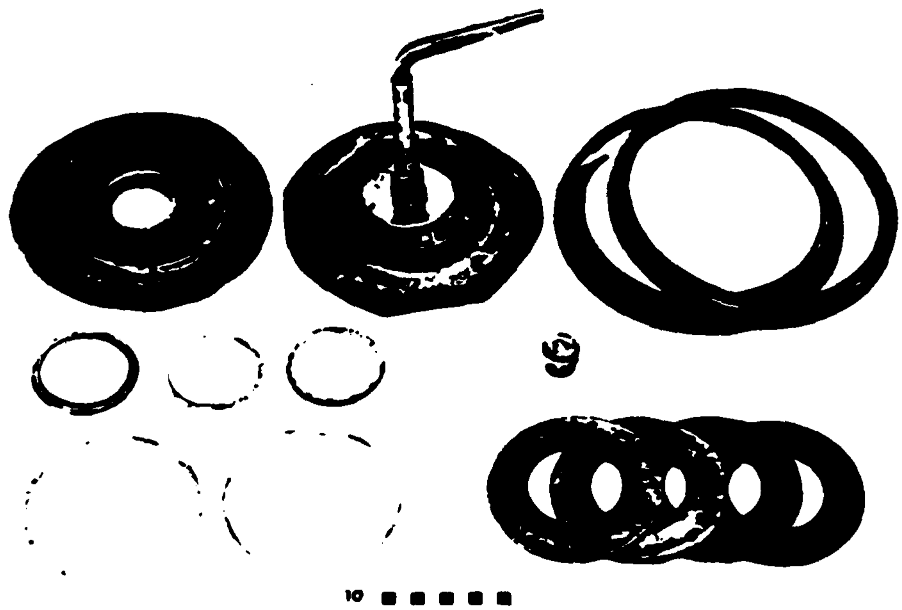


(b) Pressure Plate and Detonator Components

Figure 2.7 Damage to antitank mine in test 9.

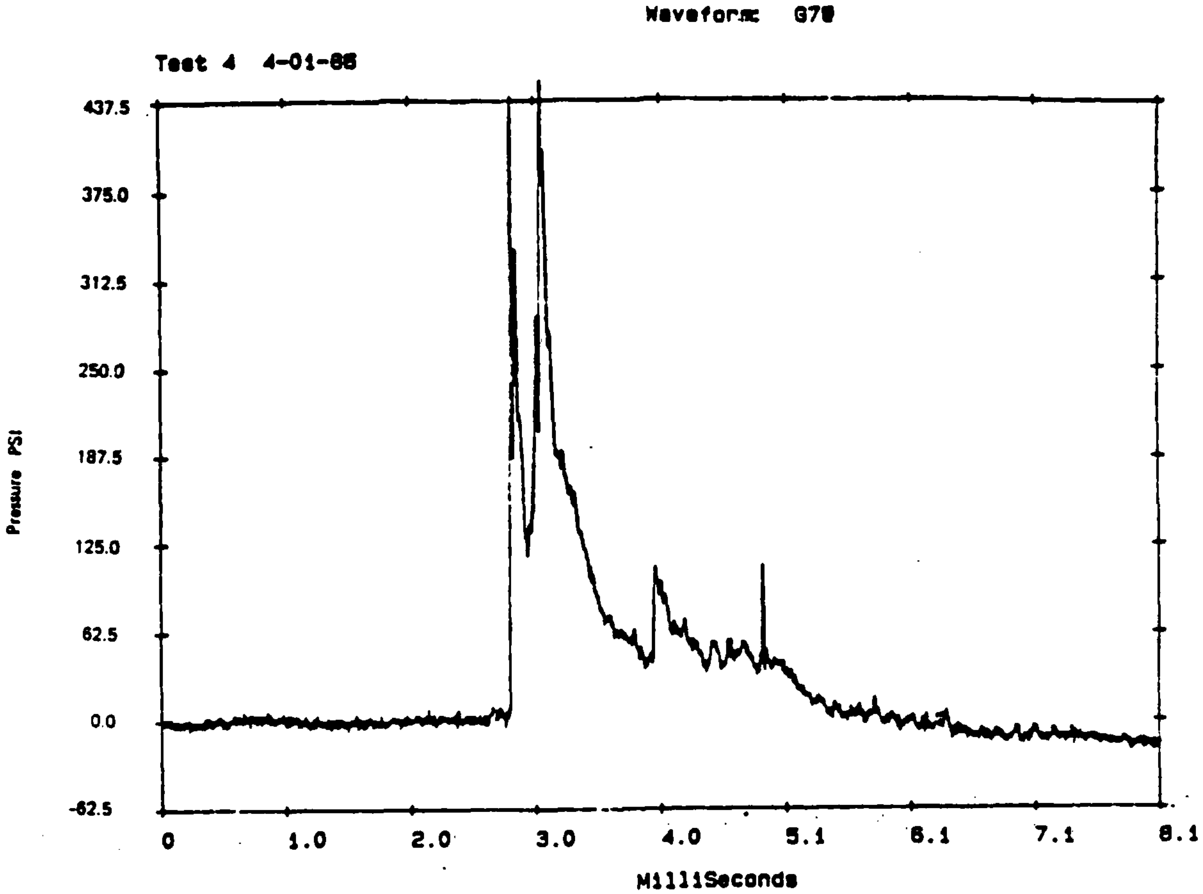


(a) Mine Casing

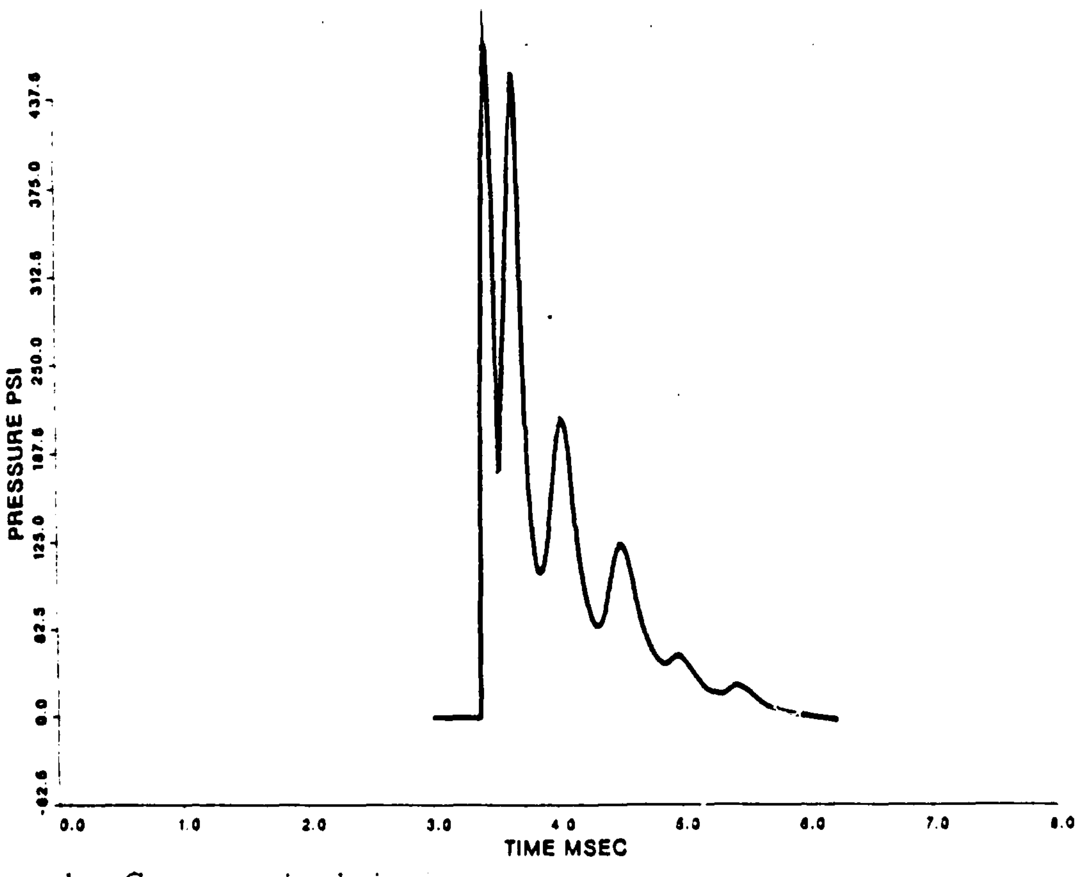


(b) Pressure Plate and Detonator Components

Figure 2.8 Damage to antitank mine in test 10.

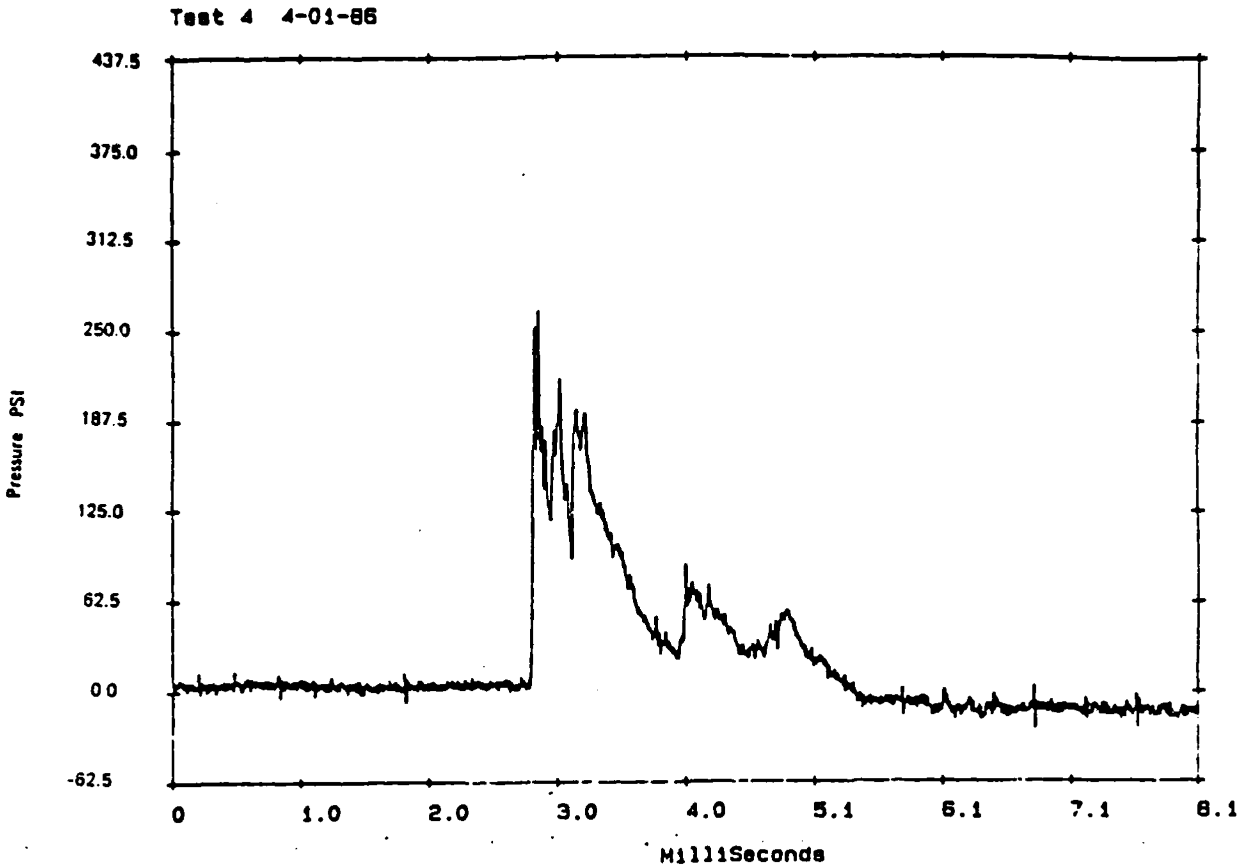


a . Experimental data

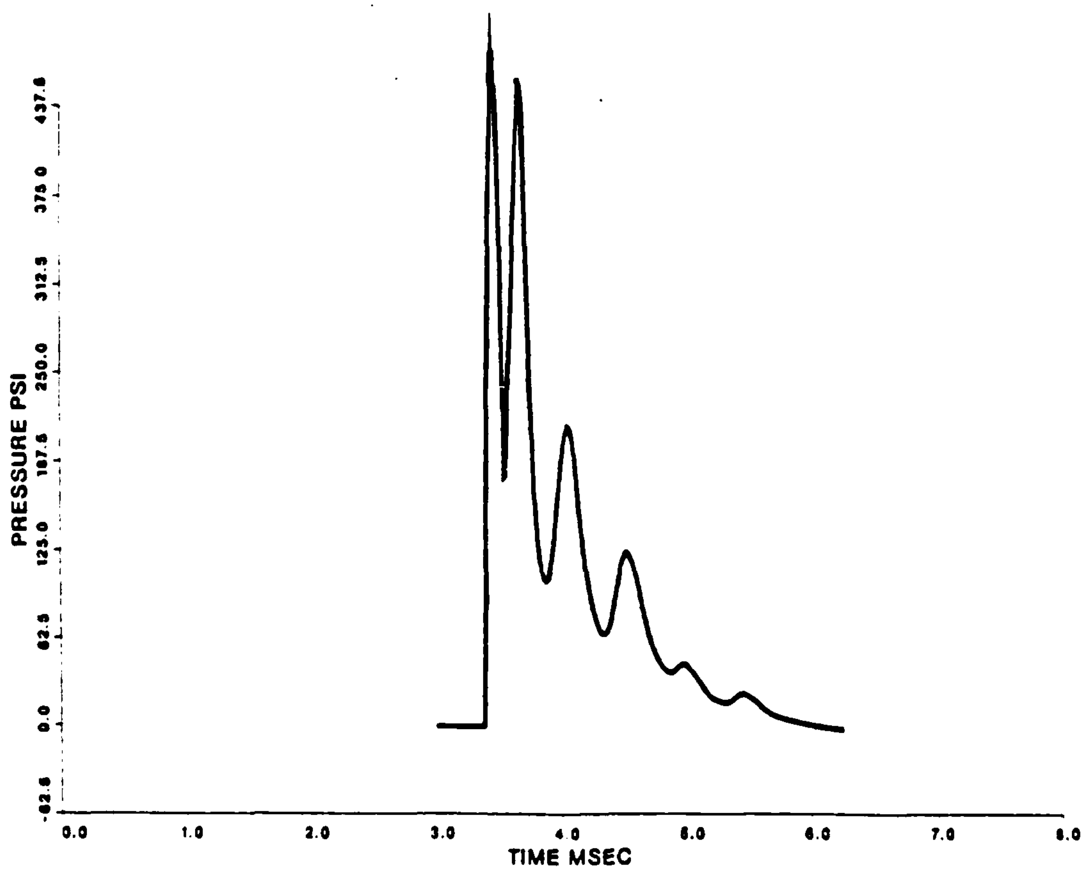


b . Computer simulation.

Figure 3.1 Comparison of data from gauge G7 (test 4) and simulation.

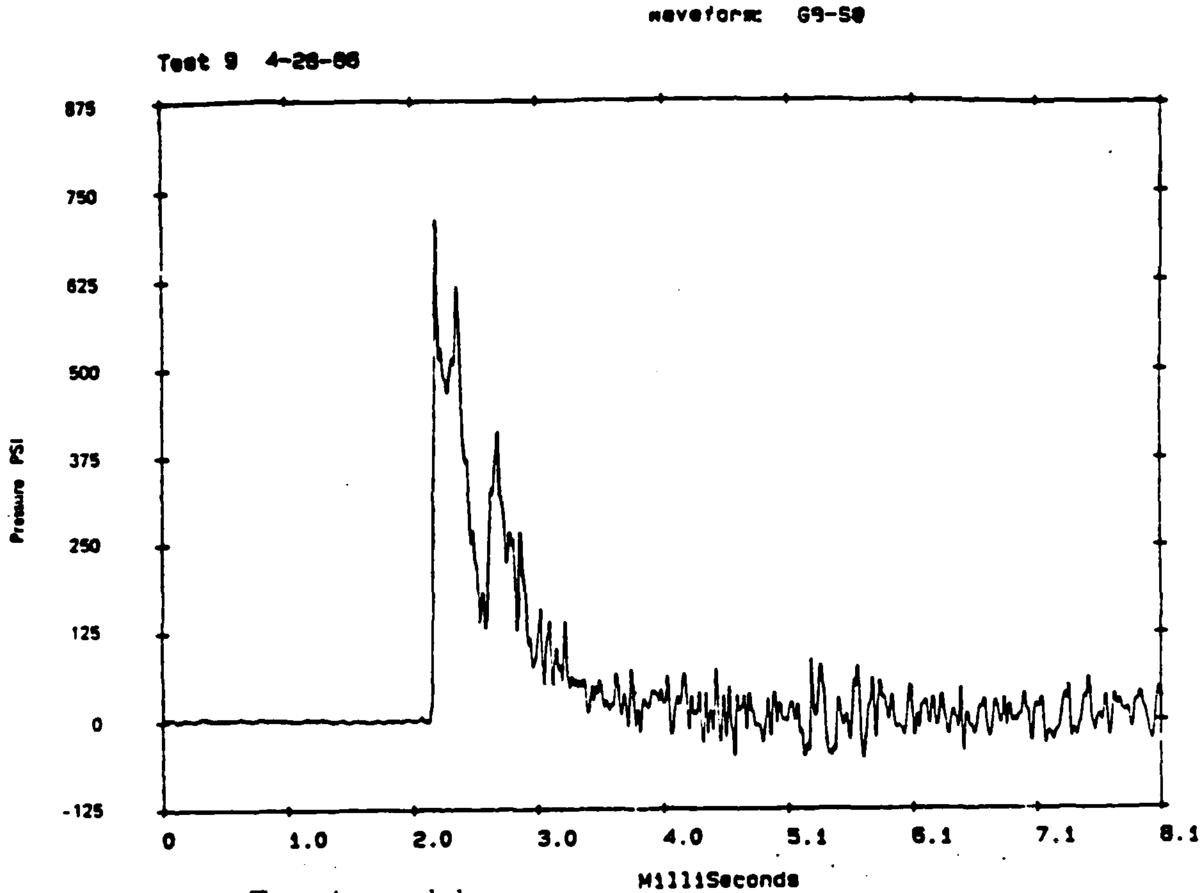


a . Experimental data

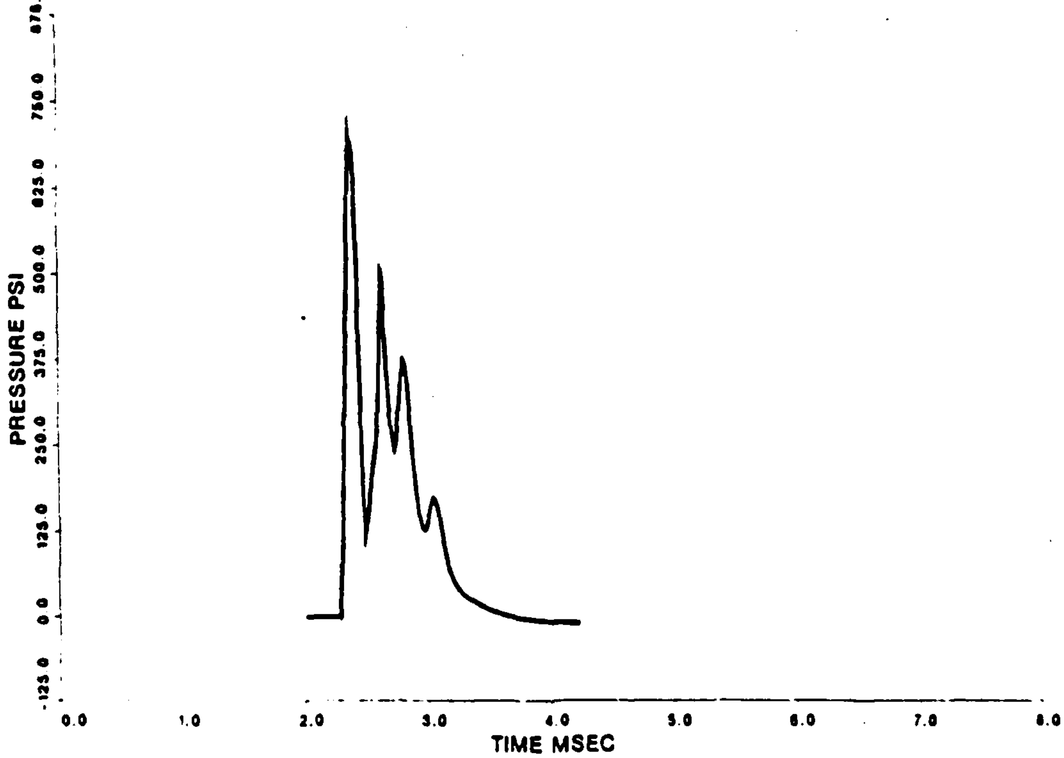


b . Computer simulation.

Figure 3.2 Comparison of data from gauge G8 (test 4) and simulation.

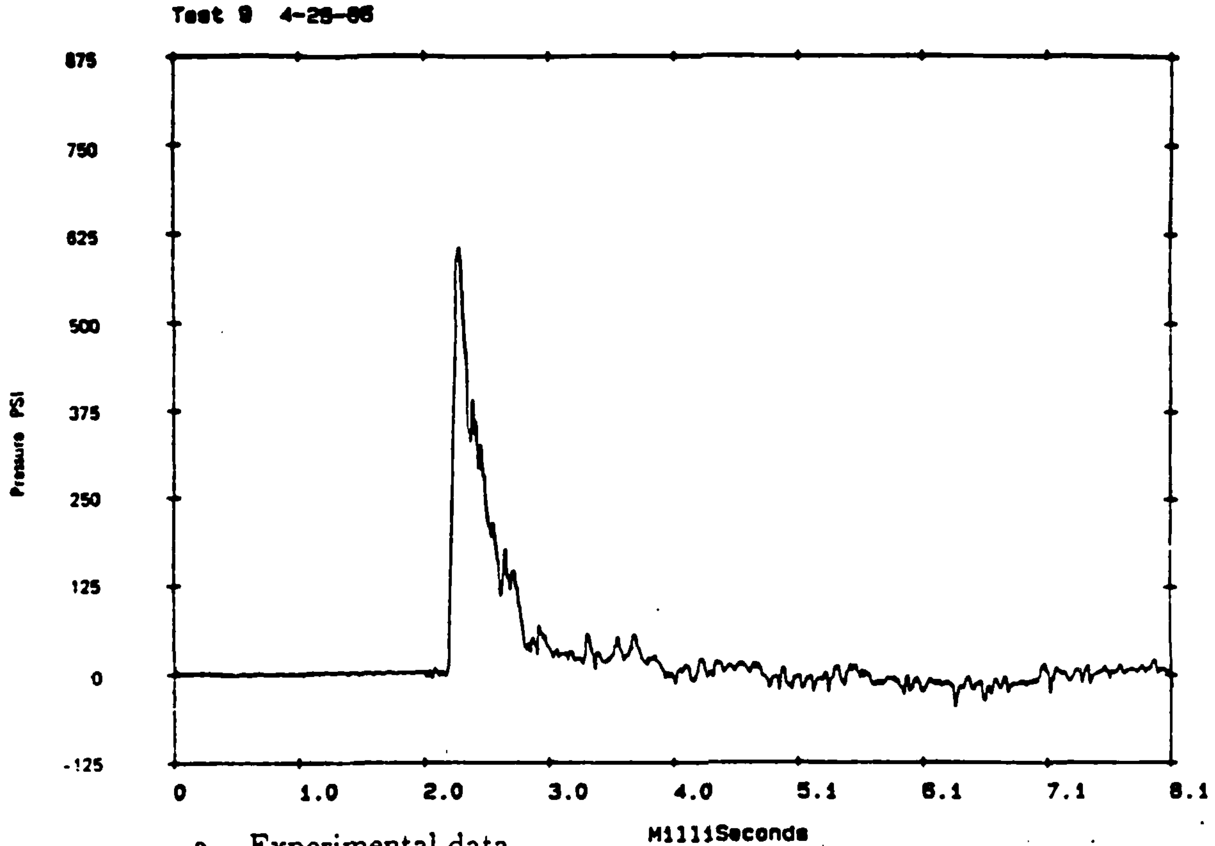


a . Experimental data

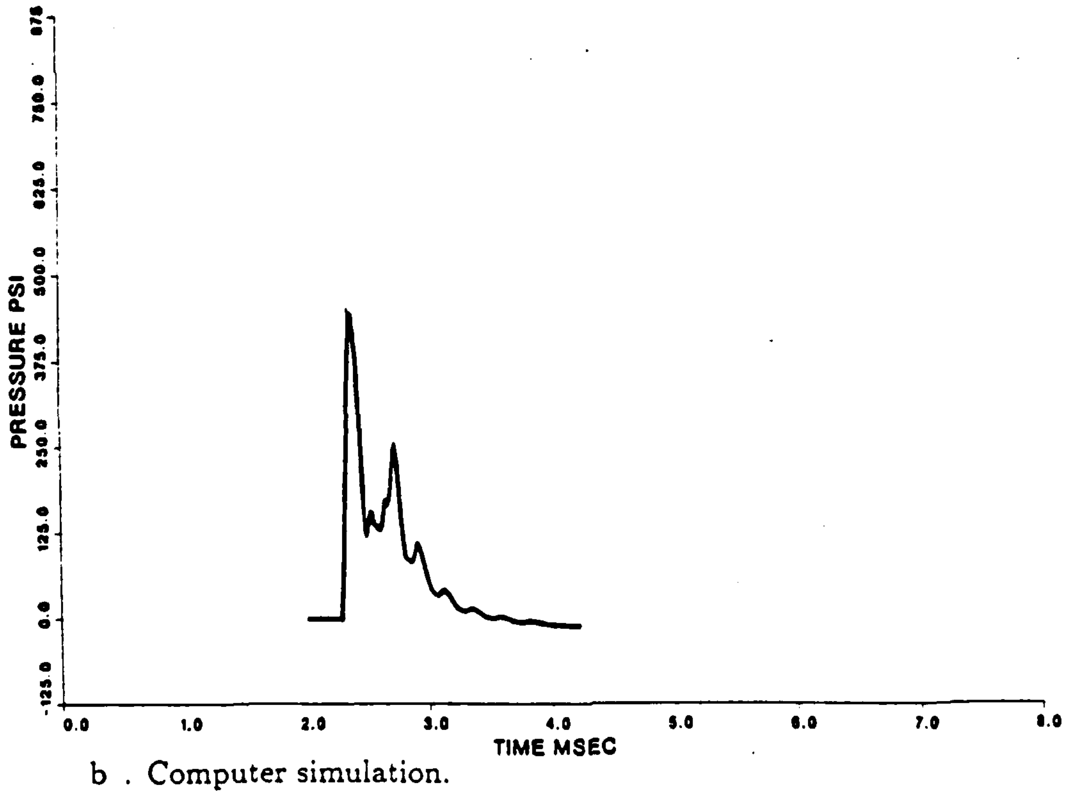


b . Computer simulation.

Figure 3.3 Comparison of data from gauge GS (test 9) and simulation.

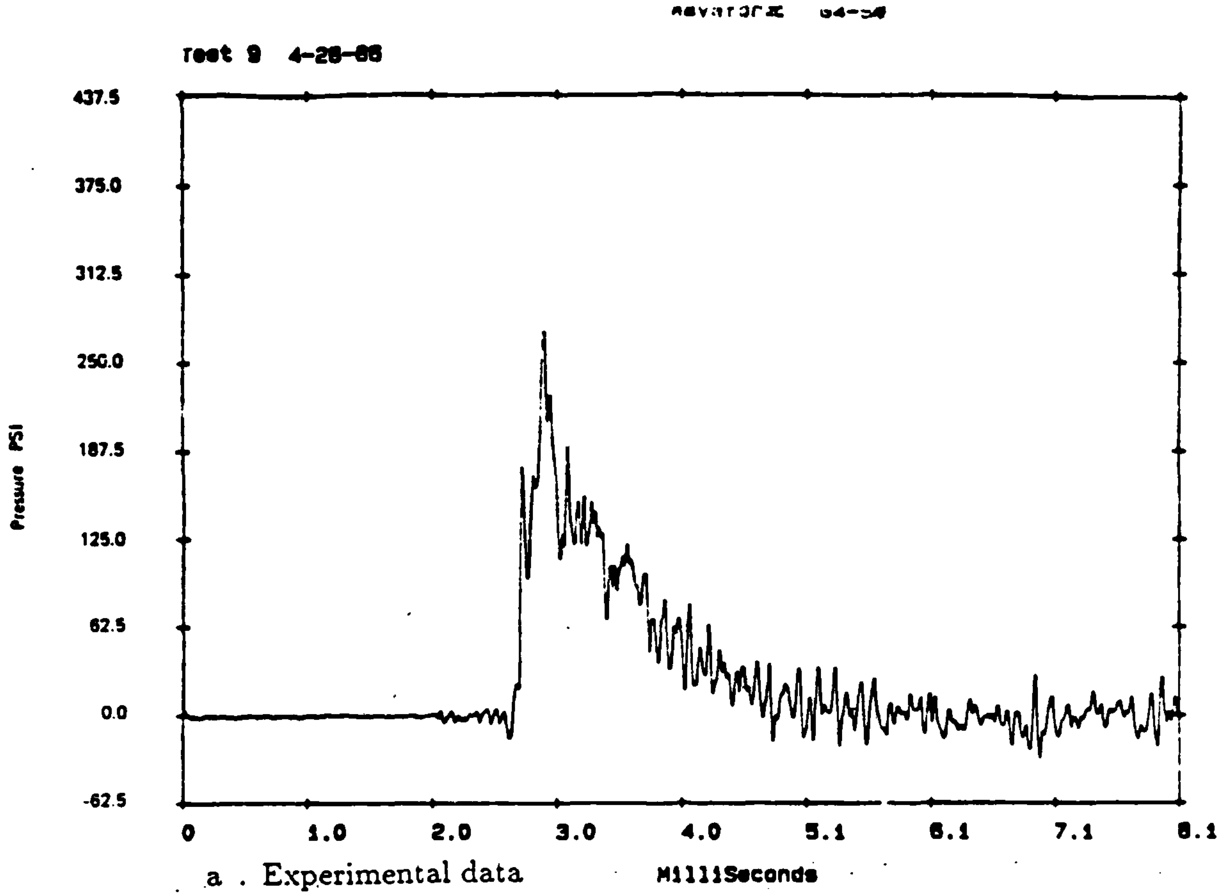


a . Experimental data

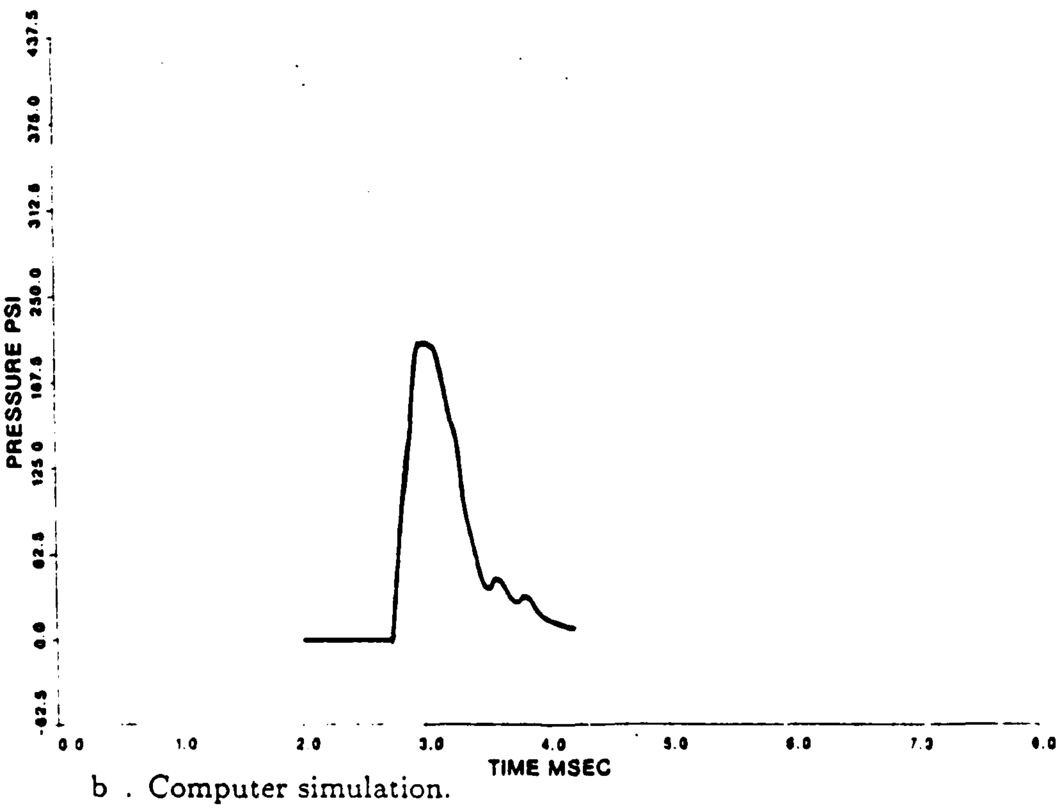


b . Computer simulation.

Figure 3.4 Comparison of data from gauge G7 (test 9) and simulation.

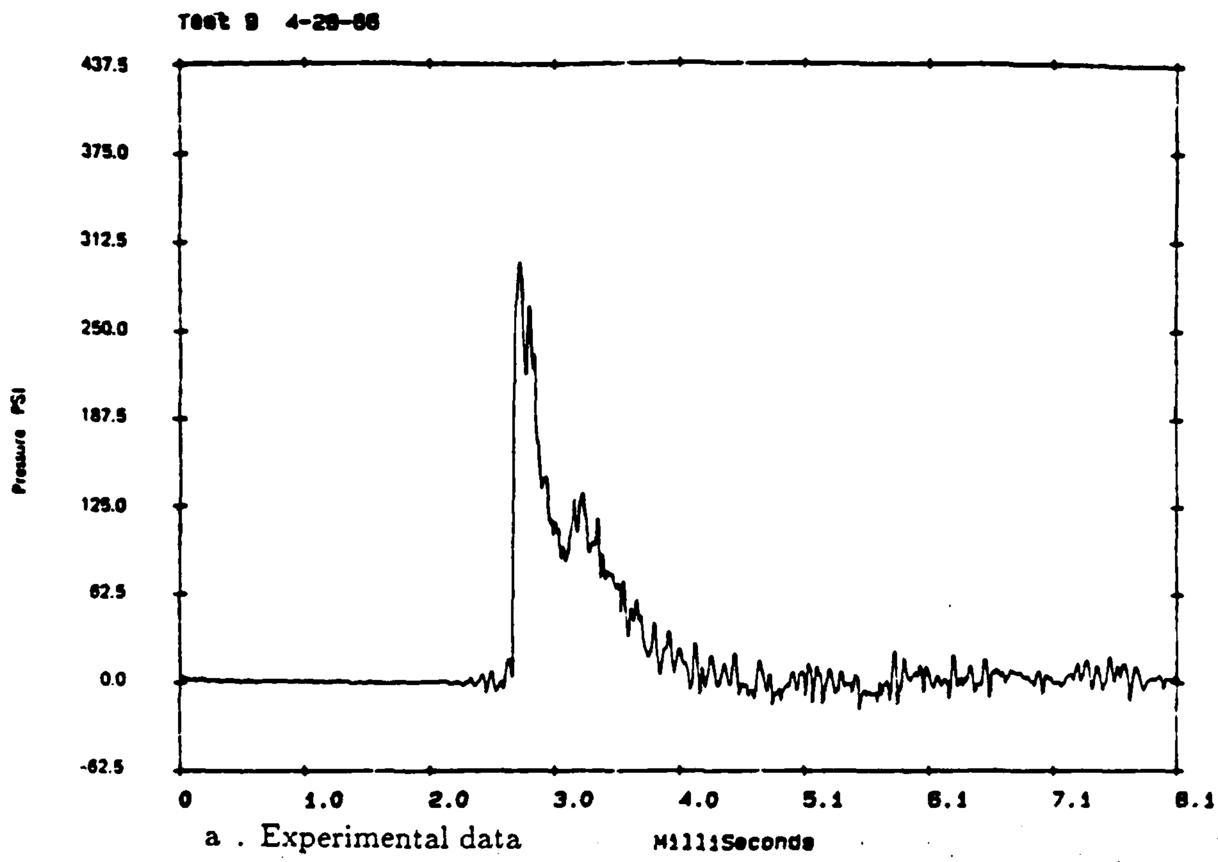


a . Experimental data

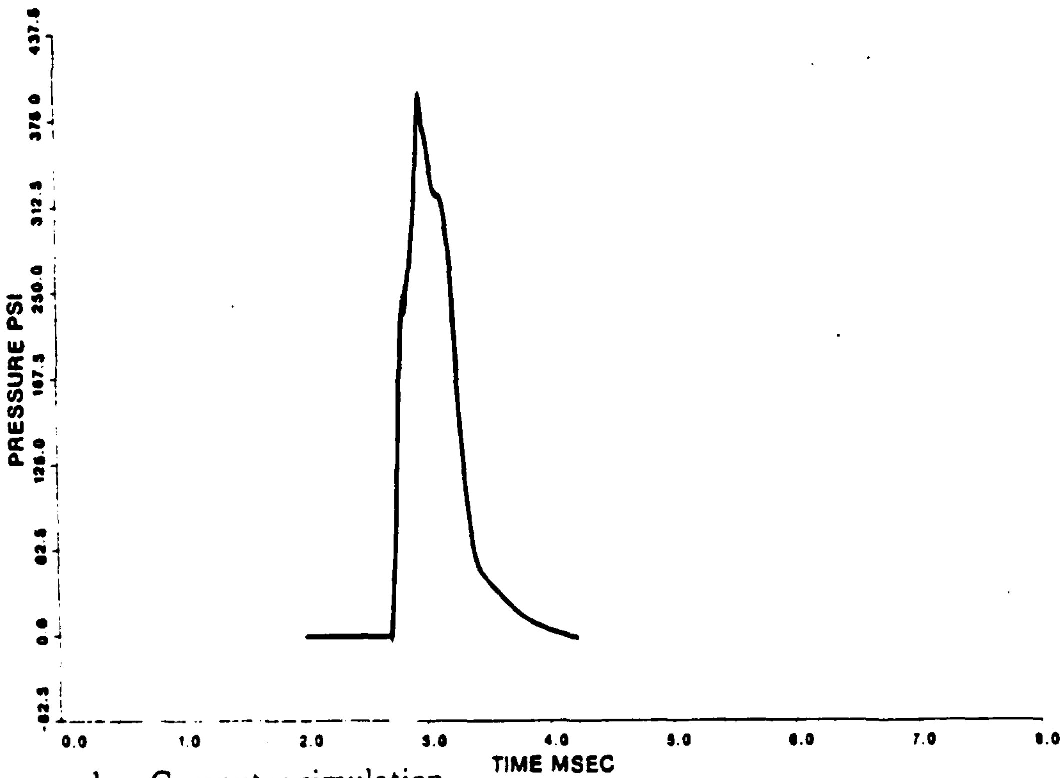


b . Computer simulation.

Figure 3.5 Comparison of data from gauge G4 (test 9) and simulation.



a . Experimental data



b . Computer simulation.

Figure 3.6 Comparison of data from gauge G5 (test 9) and simulation.

APPENDIX A

THREE-DIMENSIONAL ALGORITHM DESIGN

FOR HYDRODYNAMIC CALCULATIONS

Mark A. Fry and Pradeep Kamath

Science Applications, Inc.

1710 Goodridge Drive

McLean, VA 22102

and

David L. Book

Laboratory for Computational Physics

Naval Research Laboratory

Washington, DC 20375

ABSTRACT

A new three-dimensinal hydrodynamic algorithm has been deveoped utilizing a virtual system for data storage. The code called FAST3D solves partial differential equations aproximated by finite differences. Data are stored on y - z planes, so that only three planes are in memory at any given time. The choice of the algorithm together with the data handling system provide a highly accurate and efficient simulation tool for use on vector computers. Sample calculations are provided to illustrate the versatility of the computer code.

1. Introduction

Three-dimensional hydrodynamic algorithms must be accurate and highly efficient. Large amounts of memory coupled with large amounts of central processer time (CPU) are required for even moderately well resolved problems. FAST3D addresses these points as well as obtaining the required storage by implementing a virtual system. Details of the algorithm and sample calculations are presented.

2. Design And Implementation of FAST3D

We describe FAST3D, a code which solves the three-dimensional hydrodynamic equations for the conservation of mass, momentum and ennergy of an ideal fluid using a new leapfrog FCT algorithm. The equations advancing the dependent variables are

$$\frac{\partial \rho}{\partial t} + \nabla\cdot(\rho v) = 0; \tag{1}$$

$$\frac{\partial(\rho v)}{\partial t} + \nabla\cdot(\rho vv) = -\nabla(P - P_a) - (\rho - \rho_a)\, G(z); \tag{2}$$

$$\frac{\partial E}{\partial t} + \nabla\cdot(Ev) = \nabla\cdot(Pv) + S - L. \tag{3}$$

To calculate the pressure we employ an equation of state in the form

$$P = P(e,\rho), \tag{4}$$

where $e = E - \frac{1}{2}\rho v^2$. In Eq. (2), the ambient pressure P_a and density ρ_a are subtracted from corresponding time-dependent quantities in the source terms to avoid the "falling sky" problem; G(z), the acceleration due to gravity, is generally taken to be constant at its nominal sea-level value. S and L denote energy sources and losses.

The solution is found by numerically advancing the equations in Cartesian geometry on a rectangular variably-spaced mesh. The basic finite-difference technique is the fully three-dimensional leapfrog scheme with no time splitting. This scheme has very low dissipation, with a linear amplification matrix of unity on a uniform grid, but is subject to a weak (grid separation) instability and large dispersive errors at short wavelengths. These difficulties are overcome by using the Flux-Corrected-Transport (FCT) technique, in which the final solution is made up of the weighted average of a high-order scheme (leapfrog) and a low-order scheme, in this case upwind differencing, designed to maintain positivity (monotocity) where it is required physically. The FCT algorithm is implemented in the code by adding to the leapfrog convective terms a diffusive flux pro-

portional to the absolute velocity, plus a small velocity-independent diffusive term. If F represents any of the dependent fluid variables, the finite-difference scheme used to approximate the left-hand side of Eqs. (1)-(3) is given by

$$F^{td}_{ijk} = F_{ijk} + \frac{\Delta t}{\Delta x}\left[(Fu)_{i-1} - (Fu)_{i+1}\right] + \frac{\Delta t}{\Delta X}\, u_{i+1}\, (F_{i+1} - F_i)$$

$$- \frac{\Delta t}{\Delta x}\, u_{i-1}\, (F_i - F_{i-1}) + C(F_{i+1} - 2F_i + F_{i-1}) + \text{other two directions} \quad (5)$$

for the half of the zones which are active at any given time step, and

$$F^{td}_{ijk} = F_{ijk} + \left(\frac{\Delta t}{\Delta s}\, u_i + C\right) (F_{i+1} - 2Fi + F_{i-1}) + \text{other two directions} \quad (6)$$

for the half which are inactive. Here C is a constant used to apply extra diffusion (beyond that needed to ensure positivity) in strong-shock problems.

An antidiffusive step is added to the algorithm in timesplit fashion, removing most of the added diffusion. To do this we begin by using the transported diffused momenta and density to calculate an interim velocity in each direction:

$$\overline{u}_i = (\rho u)^{td}_i / \rho^{td}_i . \quad (7)$$

Then we calculate "raw" antidiffusive fluxes for the active zones,

$$\phi_{i-1/2} = \left(\frac{\Delta t}{\Delta x}\, \overline{u}_{i-1} + C\right) (F^{td}_{i+1} - F^{td}_i), \quad (8a)$$

$$\phi_{i-1/2} = \left[\frac{\Delta t}{\Delta x}\, \overline{u}_{i-1} + C\right] (F^{td}_i - F^{td}_{i-1}), \quad (8b)$$

and the inactive zones,

$$\phi_{i+1/2} = \left(\frac{\Delta t}{\Delta x}\, u_i + C\right) (F^{td}_{i+1} - F^{td}_i), \quad (9a)$$

$$\phi_{i-1/2} = \left(\frac{\Delta t}{\Delta x}\, \overline{u}_i + C\right) (F^{td}_i - F^{td}_{i-1}). \quad (9b)$$

Using the difference between neighboring values of the transported diffused

quanitities,

$$\Delta_{i+1/2} = F^{td}_{i+1} - F^{td}_{i}, \tag{10}$$

and the signs of the raw fluxes,

$$S_{i+1/2} = \mathrm{sign}(\phi_{i+1/2}), \tag{11}$$

we calculate the "corrected" fluxes according to the Boris-Book strong flux-limiting formula:

$$\phi^{c}_{i+1/2} = S_{i+1/2} \max[0, \min(\phi_{i+1/2}, \sigma_{i+1/2}\Delta_{i+3/2}, \sigma_{i+1/2}\Delta_{i-1/2}]. \tag{12}$$

These fluxes are then used in the antiduffusion stage to get the new values of the dependent variable:

$$F^{new}_{ijk} = F^{td}_{ijk} + \phi^{c}_{i-1/2} - \phi^{c}_{i+1/2} + \text{other two directions.} \tag{13}$$

The flux-correction part of the solution leaves just enough of the low-order scheme to guarantee monotonicity of the solutions. The driving terms [right-hand sides of Eqs. (1)-(3)] can be added in at almost any point in the calculation. Currently this is done following the transport and diffusion in Eq. (5).

The main region of the calculation employs equally spaced zones to maintain high acuracy. The grid, which does not vary in time, may be stretched in regions of little activity or near the edges to reduce the influence of boundary approximations. Either reflecting or outflow boundaries may be applied in the transverse (y and z) directions and a choice of reflecting, outflow, or periodic boundary conditions is availabe in the longitudinal direction. A variable time step is used based on the minimum CFL limit in each direction over the whole mesh, $\Delta t < \min[\Delta x/(v + c), \Delta y/(v - c), \Delta z/(v + c)]$. The momentum equation contains a gravity term. A real-gas equation of state is also available for strong shock calculations in air.

For our applications, the shocks and other unsteady phenomena being simulated are initially localized near one end of the system, conventionaly taken as the origin of the x axis. At early times it is unnecessary to update variables far from this region, so the number of active zones in this direction is $(NX)' < (NX)'_{max}$. At late times $(NX)'$ increases until the entire mesh is active.

Three-dimensional hydrodynamic calculations require large amounts of memory, typically more than is physically available on a given computer. This necessitates the use of auxiliary storage such as disk. The FAST3D algorithm uses only the neighboring values of the fluid variables to time-advance a given value. The data are stored on y-z planes, so that only three planes need be in memory at any given time during the computation (Fig. 1). Furthermore, by allowing I/O buffers for transferring planes to and from the disk, computation can be overlapped with I/O to minimize overall computer time. This implementatin of the algorithm requires two passes through the auxiliary disk file per time step: one for the leapfrog time advancement and one for the FCT correction. The parallel architecture of the Cray XMP series can be utilized here. Other passes through the disk file which do not occur at every time step may be made as needed, e.g., to produce diagnostics or restart/dump files.

The leapfrog FCt algorithm involves enough artithmetic to keep the transport part of the FAST3D code compute-bound. However, due to the explicit nature of the algorithm, the computations on the planes can be written to take advantage of vecotr hardware that exists on machines such as the Cray-1. When fully vectorized, FAST3D runs on the Cray-1 at about [illegible]μs (zone-step). The code has also been run an the Cray-XMP/12 at 15μs/(zone step).

3. Problems and Results

The code was tested on a variety of simple idealized calculations. The Riemann (exploding diaphragm) problem was used since it embodies all the gas dynamic discontinuities one finds in shock physics. Figure 2 shows the density as a function of distance some time after diaphragm rupture. Initially the density and pressure increased across the diaphragm from left to right by factors of 10 and 100, respectively. An ideal gas equation with a gamma of 1.4 was used to relate pressure to internal energy. Examination of Fig. 2 reveals an accurate simulation of the shock front moving to the left, the contact discontinuity with only a small amount of smearing, and the smoothly varying rarefaction region propagating to the right.

A more difficult problem is presented to indicate the versatility of the FAST3D code. Shock tubes with moderate shock strengths corresponding to over pressures of approximately three atmospheres have been used to investigate the physics of shock diffraction in the pressence of heated layers and obstacles. Helium gas layers are generally used to simulated the higher temperature gas along the wall of the tube. Rigid obstacles are constructed and fixed downstream from the diaphragm. When the shock wave encounters the heated layer (helium) it begins to propagate faster and creates an additional contact surface (discontinuity in density but not pressure). Furthermore, the shock is inclined, moving upward, as one moves away from the shock tube wall. Figure 3 shows the density and pressure contours in the plane of symmetry (x - z axis). Note the "toeing out" of the shock front into the heated layer. When an additional discontinuity such as an abstacle is placed in this flow field, a more complex flow is produced. Three-dimensional graphics provide a unique way to view what is taking place. Graphics programs have been created with the same virtual data handling system as found in FAST3D. The results are show in Fig. 4. A series of four different density level contour plots, all at the same

time, provide a look into the features of interest in the calculation. Starting from left to right, one first sees a density level of $\rho = 5.0 \times 10^{-}$ which represents the helium layer. The intersection of the three axis is the origin for the cartesian frame of reference. The shock wave is moving from back to the front. The axis parallel to horizontal is the y axis and the vertical axis is the z axis. Second from left is the density level, $\rho = 2.0 \times 10^{-3}$, which shows the shock impinging on the rigid structure and the diffraction of the shock over the structure, respectively.

Current large-scale scientific computers provide insufficent central memory for even moderate simulations. The FAST3D code combines a state-of-the-art numerical algorithm with storage designed for vector computers. Additional speedups are possible by taking a account of the parallel features in the code. The result is the ability to model gas dynamic systmes which previously were too complex and to achieve accuracies not otherwise available.

Fig. 1. Representation of the data storage system used in FAST3D.

Fig. 2. Density and Pressure vs. distance for the Riemann problem. The vertical line in the middle indicates the position of the diaphragm.

Fig. 3. Density and Pressure contours in the x - z plane of symmetry. The shock is advancing from left to right. Note the thermal layer along the bottom in the density figure.

Fig. 4. Three-dimensinal representation of the shock on structure in a heated layer. Z axis is vertical to page, y axis is horizontal, and x axis moves out of page. The shock is moving from back to front. The figure at far left is contoured with density = 5×10^{4} g/cm^3. The middle is contoured with density = 2×10^{-3}. Lower right is density = 20 and upper right is density = 1.0×10^{-3}.

THREE — DIMENSIONAL OUT-OF-CORE CALCULATIONS

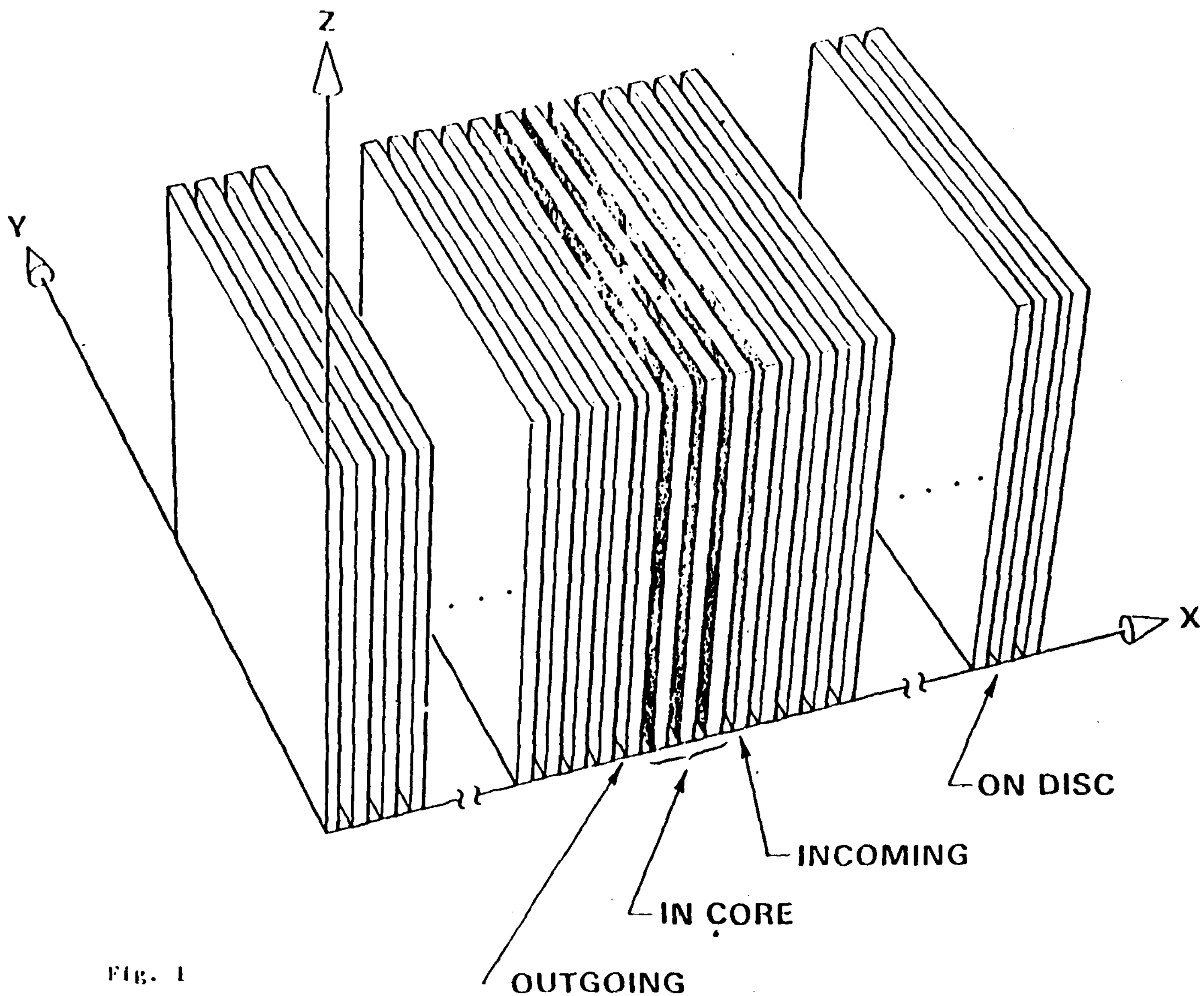


Fig. 1

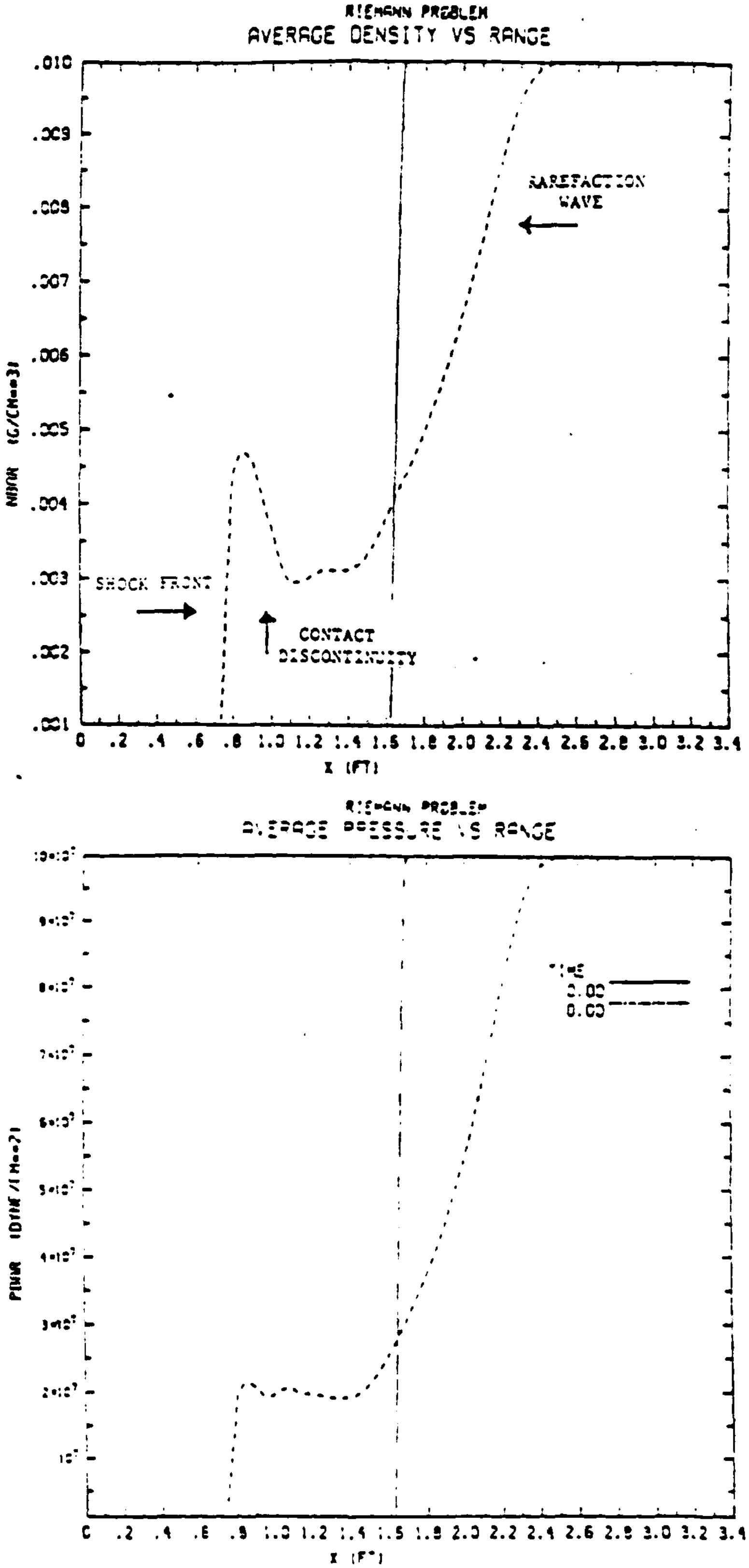

RIEMANN PROBLEM
AVERAGE DENSITY VS RANGE
RAREFACTION WAVE
SHOCK FRONT
CONTACT DISCONTINUITY
X (FT)
RIEMANN PROBLEM
AVERAGE PRESSURE VS RANGE
TIME
0.00
0.00
X (FT)

Fig. 2

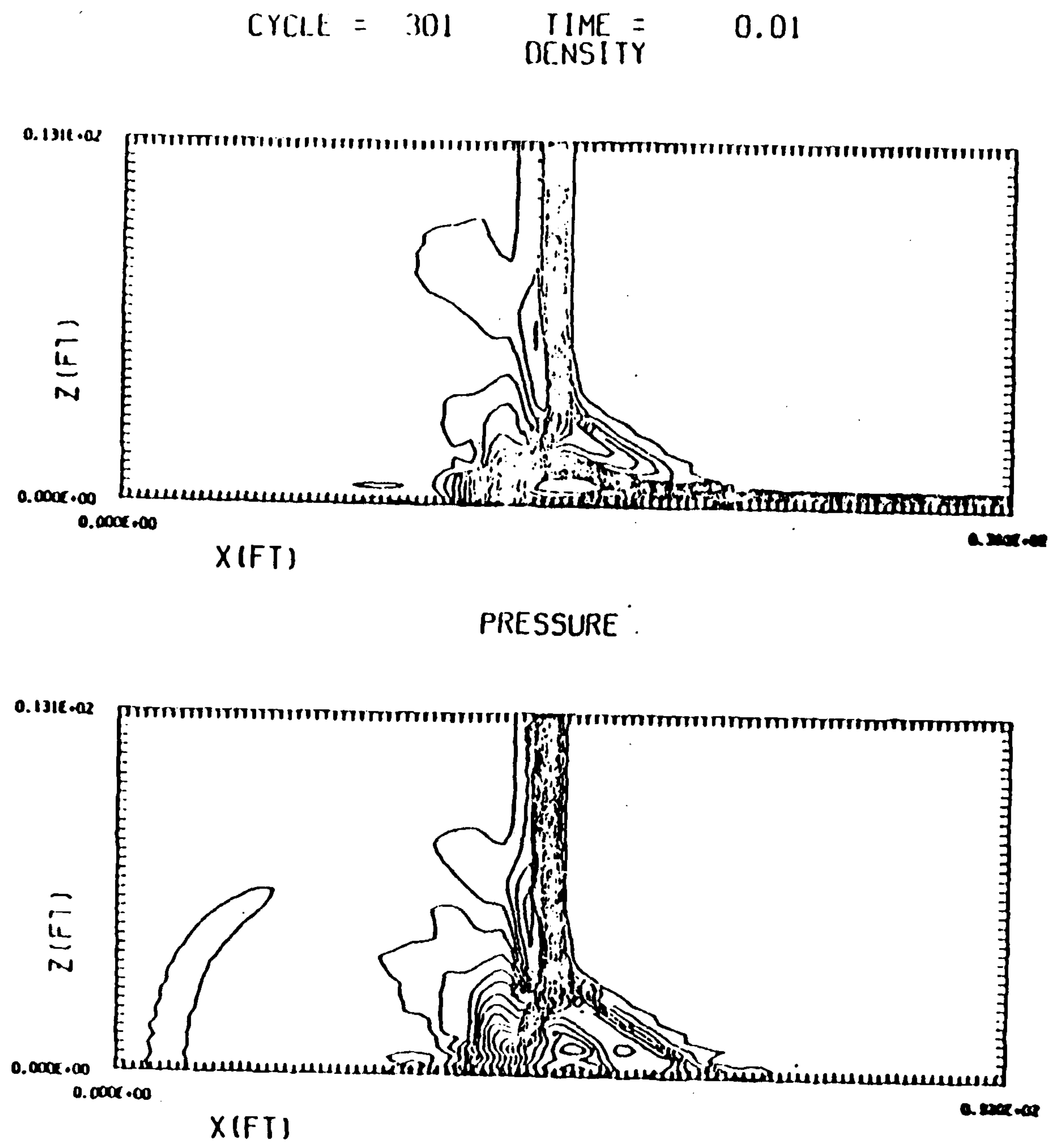

CYCLE = 301 TIME = 0.01
DENSITY
0.131E+02
Z(FT)
0.000E+00
0.000E+00
X(FT)
PRESSURE
0.131E+02
Z(FT)
0.000E+00
0.000E+00
X(FT)

Fig. [illegible]

THREE - DIMENSIONAL GRAPHICS

Shock on Structure in Heated Layer

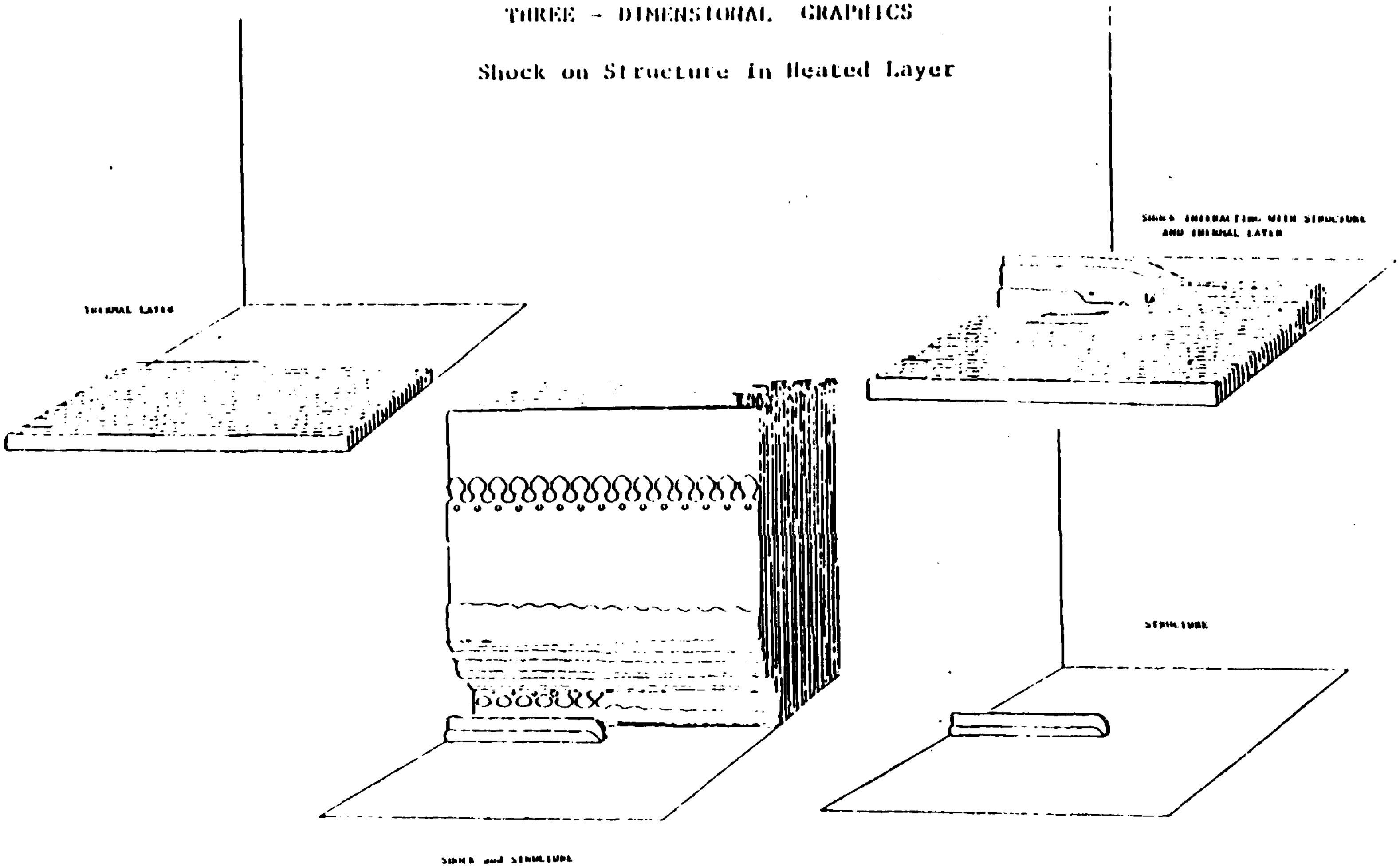


Fig. 4

APPENDIX D

COMPILATION LISTING OF

FLIK 3D

GRAPHICS CODE

THE KORAN OF FLIK3D

Flik3d, or Nplot3d as its current incarnation is named, is a graphics package created to plot the results of airblast simulations in 3 dimensions. The program is written in FORTRAN, and makes use of two different graphics libraries: NCAR (from the National Center of Atmospheric Studies) and DISSPLA (put out by ISSCO graphics software).

In its current form, nplot3d is capable of plotting fourteen different graphs, both contour and profile, plus history graphs showing the changes in various quantities over time. It can make use of calculations on both regular and irregular grids, showing all or part of a viewing window, in color or black and white.

The information specifying the plots to be drawn is read in from the file ndat, using namelists qdata and histry. The raw data to be plotted is read in in a series of dump files. The names of the dump files are listed in the file ndump in the order that they will be plotted. That is, ndump might contain the list

1 n20a0001
2 n20a0002
3 n20a0003
.
.
.
12 n20a0012

Most of the information on the plot window, slices to be plotted, and so forth is in the namelist gdata. This namelist contains the following values:

nslice -- this is the number of "slices" through the calculated region which are to be plotted. Usually up to three planes x-y, y-z, and x-z plane.

msect -- this is an array (presently from 1 to 3) of integers, holding the planes which are to be plotted. That is, if msect(1)=1, then the x-y plane will be the first to be plotted. Nplot3d will plot the first nslice slices in msect.

mswich -- this is an array (14 by 3) of integers. Each value in mswich represents one plot. If the value of mswich(m,n) is zero, graph m will NOT be plotted for slice n. The graphs are, in order:

1 density profile (species 1, 2, and 3)
2 energy profile
3 dynamic pressure profile
4 pressure profile
5 renormalized density profile (scaled to ambient)
6 renormalized pressure profile (scaled to ambient)
7 z velocity (at right angles to current plane)
8 density contour (renormalized, scaled to ambient)
9 energy contour
10 pressure contour
11 mach number contour
12 velocity vector plot
13 trace particles plot (used to track small particles)
14 grid (the actual irregular grid used in the calculations)

For the profile plots, mswich's value specifies the profile to be drawn. If mswich is positive, the profile is plotted versus distance in the horizontal direction; if negative, in the vertical direction. The absolute value of mswich gives the value of the vertical or horizontal distance. For

example, if mswich(1)=36, that means "plot density vs. distance in the z direction at x=36."

ilef -- ilef is an array (dimension 3) of integers, each giving the left most boundary of the region to be plotted in the x direction for a particular slice.

irig -- irig, similarly to ilef, gives the rightmost boundary in the x direction for each slice.

jfnt -- jfnt gives the "frontmost" boundary of the region to be plotted in the y direction for a particular slice.

jbak -- jbak gives the "backmost" boundary of the region in the y direction.

kbot -- kbot gives the lower boundary of the region to be plotted in the z direction for each slice.

ktop -- ktop gives the upper boundary of the region to be plotted in the z direction for each slice.

nfiles -- gives the number of dump files to be plotted.

incore -- incore is a logical which specifies whether the calculations will be made entirely in core or not.

color -- this logical determines whether the contour plots will be plotted in color or black and white.

lstat -- a logical, stating whether or not there are sensor stations present.

interp -- this logical should be true if the data was calculated on an uneven grid. The values will be interpolated onto an even grid for plotting.

dmin -- this is the minimum contour level for the contour plot of density. If dmin=dmax=0 then the program will choose its own, rounded minimum.

dmax -- this is the maximum contour level for the contour plot of density. See dmin.

displa -- displa is a logical which determines which graphics library will be used. If displa=true, then the graphs will be plotted using DISSPLA. Otherwise, NCAR will be used.

nbu -- this is the number of bursts for which the particles should be plotted.

darkhd -- this logical is used in conjunction with the displa logical. If displa and darkhd are both true, then headings will be printed in a dark, bold type (swissbold). If darkhd is false, the headings will be in a simple line font. All NCAR headings are simple.

ilines -- this is used with the grid plot. Every ILINESth vertical line will be drawn.

jlines -- this is used with the grid plot. Every JLINESth horizontal line will be drawn.

There are a number of particular problems to look out for in modifying this code, as well as in running it. One particularly subtled problem is that, in compiling and loading the code, the DISSPLA library discft MUST be loaded first. The proper command would then be

```
rcft i=nplot3d lib=(discft,ncarcft,cgscft,cftmath)

                or

cft i=nplot3d
ldr lib=(discft,ncarcft,cgscft,cftmath)
```

If this is not done, then conflicts arise between NCAR and DISSPLA commands with the same names.

Also, in running this program, only use darkhd=true if your object is to produce presentation quality plots. These plots are high in quality, but consume almost ten times as much CPU time as plots with darkhd=false. If one is concerned with working plots, simply for your own reference, one should use NCAR, or DISSPLA with darkhd=false. And, if you run off plots with dark headings, print them out at the electrostatic printer or microfilm

utility at Los Alamos. Otherwise, producing plots is too time consuming. Old age comes fast, sitting in front of a tektronix terminal.

Within the code itself are some peculiarities that must be watched out for. If you, for any reason, wish to add or take away plots, extensive changes must be made. First, change the declared dimension of mplot and mswich EVERYWHERE in the program. Secondly, look for the do loops in conv3d and prof3d. They are of the form

```
do xxx ip=m,n
```

These do loops, and the assigned gotos that go with them, must be altered for the programs to function properly. Also, after the section of code which actually draws the plots, there is a line which checks to see if other species of density are to be drawn. This line, too, makes reference to ip.

```
if (ispec .le. nspec .and. ip-7 .eq. 1) go to 50
```

The particle plot is another potential trouble source. At present, it is hard-wired to trace the particles from up to 2 bursts. In order to increase the number of bursts, one must alter several declarations at the beginning of Conv3d. The data statements assigning ichar and imark must be changed; the parameter nburst must be increased.

Another frequent change will have to be made in the parameter statements. Every time the grid is changed, you must change the parameters nnnx, nnny, nnnz, ndm1, ndm2, and nhv. Nnnx, nnny, and nnnz are the dimensions of the 3D grid used in calculations, and ndm1 and ndm2 are the dimensions of the 2D slices.

Several general points to mention:

1. Parameters are declared in many places in this code. Make sure to change them everywhere.
2. Line numbering is erratic at best in conv3d and prof3d. If you must add a line number, check to make sure that it is not already used.
3. Similarly, those routines make use of many small variables, in small do loops and such. All such variables are listed in the following listing, after the routines in which they occur. Before you add a variable, check to make sure that it has not been used for something else.
4. Two different space arrays have been provided for use. If you need to modify some variable (pressure, density, or whatever), USE THE SPACE ARRAYS! DO NOT OVERWRITE THE MAIN ARRAYS! These arrays are used in other calculations, and changes will cause the other plots to be invalid.
5. This code is long, tangled, and obscure. The comments are sometimes inadequate. Be careful, it is easy to miss some needed changes and cause errors that are very difficult to find.

The output of this program will be a cgs metafile titled "plot." This metafile can be viewed on a tektronix terminal using pscan or ptekt, or printed out on the electrostatic printer using pesp, or printed to film using pfilm. Good luck with this program -- you may well need it!

Todd Brun
19 August 1986

Errata

At the end of every plot file will be a blank page, present for reasons unknown. Ignore it, it's harmless.

Every velocity vector plot will be followed by a blank page. This is a

small bug (a feature, now that I've documented it) which is caused by the use of the DISSPLA and NCAR calls in combination, and cannot be easily corrected. As it is also harmless, I have not tried to correct it. Thus, if one requests all fourteen plots for two dump files, one slice each, the plot file will contain 31 pages 28 plots, 2 blank pages after the velocity vector plots, and one blank page at the end. If you try it and get 3 pages, something is wrong

```
      program flik3d (tty,output=tty)
c-----------------------------------------------------------------
c     main program for FAST3D graphics
c-----------------------------------------------------------------
c         Flik3d is a graphics package designed to print several different
c  plots from information stored in a dumpfile, produced by an airblast
c  simulation program.  These plots are produced by two major routines,
c  Conv3d (which produces contour plots, velocity vector plots, particle
c  plots, and a template grid), and Prof3d (which produces various profile
c  plots), supported by a number of other routines.
c         The graphics are drawn using two different graphics libraries:
c  DISSPLA (by ISSCO) and NCAR.  The library used is controlled by a
c  logical switch 'displa', which is read in in the namelist gdata and
c  passed in the common block grfcom.
c         The specifications of the plots are input from the file ndat.  Most
c  of these are passed in the namelist gdata.  The array mswich specifies
c  which plots are to be drawn, while the switch displa determines which
c  library is to be used.  Nslice holds the plane which is to be plotted:
c  x y, y z, or x z.  For full explanation of the ndat file, see the
c  Flik3d Koran.
c  To run correctly, the source code must be compiled and loaded correctly.
c  The proper format is
c     rcft i=uplot3d lib=(discft,ncarcft,cgscft,cftmath)
c  The discft library must be loaded first, in order for the program to run
c  correctly.
c         Many parts of this code are hardwired; they cannot be changed
c  easily without altering the code in several places.  Several examples
c  are listed in the Koran.  Several important ones:  the number and order
c  of the plots is controlled by the array mplot.  If more plots are added,
c  be sure to alter mplot's dimension EVERYWHERE.  Also, one must alter
c  a number of do statements (of the form do xxx ip=yyy,zzz) which control
c  the order of plot calls.
c         Furthermore, the size of the page and the subplot area is set, in
c  the routine Conv3d.  Great care should be exercised in changing them, as
c  this will require that one re-scale a number of other calls, such as
c  the SET calls which precede the velocity vector plot.
c         Also, the line numbering is irregular, at best.  Check to make sure
c  that you have not duplicated any existing line numbers.
c         Good luck with these graphics!!
      character *40 label
      character *8 readf, printf, namef, restf
      data readf /'ndat'/, printf /'nout'/, namef /'ndump'/
c------------------------- station arrays ------------------------
      parameter (mxx=1, nsta=1)
      parameter (mdmp=1, mdxx=mdmp*mxx)
      logical lstat
      real rhs(mxx,nsta), vls(mxx,nsta), gms(mxx,nsta)
      real prs(mxx,nsta), tme(mxx), xs(nsta), ys(nsta), zs(nsta)
      common /stats/ lstat, rhs, vls, gms, prs, tme, xs, ys, zs, nxx
      real tim(mdxx), psta(mdxx,nsta), dsta(mdxx,nsta)
      real star(mdxx,nsta)
      pointer (ipstar,star)
c------------------------- local declarations --------------------
      parameter (npar=3)
      logical interp(npar), incore, lintp, colour, ladj, displa
      logical darkbd
```

```
 17.          integer mswich(14,npar), msect(npar), ilef(npar)
 18.          integer irig(npar), jfnt(npar), jbak(npar), kbot(npar)
 19.          integer ktop(npar), mplot(14), ilines(npar), jlines(npar)
 20.          namelist /qdata/ nslice,msect,mswich,ilef,irig,jfnt,jbak,
           1  kbot,ktop,nfiles,incore,colour,lstat,interp,dmin,dmax,
           2  displa,nbu,ilines,jlines,darkbd
 21.          namelist /histry/ ista,jsta,ichose,iadj
 22.          common /qrtcom/ mslice, imin, imax, jmin, jmax, kmin, kmax,
           1  mplot, lintp, colour, dmin, dmax, displa, il, jl, darkbd
 23.          common /nbur/ nbu
 24.          data ilines, jlines /npar+1, npar+1/
      c
      c    open files.
 25.          open (5,file=readf,status='old')
 26.          open (6,file=prntf,status='new')
 27.          open (7,file=namef,status='old')
      c
      c    read in namelist data.
      c    this data specifies the types of plots to be drawn.
 28.          read (5,qdata)
      c
      c    initialize graphics and NCAR or DISSPLA.
 29.          call gplot (1hu,4hplot,12)
 30.          if (displa) then
 31.            call libdisp
 32.          else
 33.            call libncar
 34.          endif
      c
 35.          if (lstat) go to 120
      c
      c    plot contours and velocity vectors.
      c    *********************************
 36.          do 110 if=1,nfiles
 37.          read (7,270,end=110) restf
 38.          call restrt (istep,incore,restf,label)
      c
      c    count number of characters in label.
 39.          nlabel=0
 40.          do 10 i=1,40
 41.          nlabel=nlabel+1
 42.          if (label(i:i) eq '$') go to 20
 43.   10     continue
 44.   20     continue
 45.          if (incore) call readin
      c
      c    process one slice at a time.
      c    the data controlling the appearance of the plots is read in from
      c    the file ndat in namelist qdata.
 46.          do 100 islice=1,nslice
 47.          mslice=msect(islice)
 48.          imin=ilef(islice)
 49.          imax=irig(islice)
 50.          jmin=jfnt(islice)
 51.          jmax=jbak(islice)
 52.          kmin=kbot(islice)
```

```
 53.              kmax ktop(islice)
 54.              il=ilines(islice)
 55.              jl jlines(nslice)
 56.              lintp interp(islice)
 57               do 30 ip=1,14
 58.       30     mplot(ip)=mswich(ip,islice)
       c
       c      fill the plot arrays and plot.
 59.              go to (40,60,80), mslice
       c
       c      X Y slice.
       c      Reads in the data and calls the routines to draw profile
       c      and contour plots.
 60        40     do 50 kk=kmin,kmax
 61               call datain (imin,imax,jmin,kk)
 62.              call prof3d (istep,restf,label,nlabel,kk)
 63.              call conv3d (istep,restf,label,nlabel,kk)
 64.       50     continue
 65               go to 100
       c
       c      Y Z slice.
       c      Reads in the data and calls the routines to draw profile
       c      and contour plots.
 66        60     do 70 ii=imin,imax
 67.              call datain (ii,ii,jmin,kmin)
 68               call prof3d (istep,restf,label,nlabel,ii)
 69               call conv3d (istep,restf,label,nlabel,ii)
 70        70     continue
 71.              go to 100
       c
       c      X Z slice.
       c      Reads in the data and calls routines to plot profile
       c      and contour plots.
 72        80     do 90 jj jmin,jmax
 73               call datain (imin,imax,jj,kmin)
 74               call prof3d (istep,restf,label,nlabel,jj)
 75               call conv3d (istep,restf,label,nlabel,jj)
 76.       90     continue
 77.      100     continue
       c
       c      close dump
 78.              call filefc
 79       110     continue
 80               go to 260
       c
       c      plot station time histories
       c      ****************************
 81       120     continue
 82               read (5,histry)
 83               kt 0
 84               kount 0
 85               kfile 0
 86               do 210 if=1,nfiles
       c
       c      read dump file name and read data from dump file.
 87               read (7,270,end 210) restf
```

```
 88            call restrt (istep,incore,restf,label)
 89            call filetc
 90.           kfile=kfile+1
         c
         c     determine number of characters in label.
 91            nlabel=0
 92.           do 130 i=1,40
 93            nlabel=nlabel+1
 94.           if (label(i:i).eq.'$') go to 140
 95      130   continue
 96      140   continue
         c
         c     time
 97.           do 150 i=1,nxx
 98      150   tim(i+kt)=tme(i)
 99.           kt=kt+nxx
         c
         c     patch time histories.
 100.          k=kount
 101           do 200 ksta=ista,jsta
 102.          istat=ksta-ista+1
 103.          go to (160,180,160), ichose
         c
         c     static pressure.
 104.    160   do 170 i=1,nxx
 105.    170   psta(i+k,istat)=prs(i,ksta)
 106.          if (ichose.eq.1) go to 210
         c
         c     dynamic pressure.
 107.    180   do 190 i=1,nxx
 108     190   dsta(i+k,istat)=0.5*rhs(i,ksta)*vls(i,ksta)*vls(i,ksta)
 109.    200   continue
 110.    210   kount=k+nxx
         c
         c     plot time histories.
 111.          do 250 ksta=ista,jsta
 112.          istat=ksta-ista+1
 113           xstat=xs(ksta)
 114           ystat=ys(ksta)
 115.          zstat=zs(ksta)
 116.          length=kt
 117           tmax=tim(kt)
 118.          if (ladj) call adjust (tim,psta(1,istat),kt,tmax,length)
 119           go to (220,230,220), ichose
         c
         c     static pressure.
 120     220   ipsta=loc(psta)
         c     separate calls are made to label axes with DISSPLA and NCAR
 121           if (displa) then
 122.           call xname ('TIME (SEC)',10)
 123.           call yname ('PRESSURE (DYNES/CM**2)',22)
 124.          else
 125.           call anotat(11htime, sec $,22hpressure, dynes/cm**2$,1,1,0,0)
 126.          endif
 127           icall=1
 128           go to 240
```

```
              c
              c     dynamic pressure.
      129.   230      ipstar=loc(dsta)
      130.            if (displa) then
      131.             call xname ('TIME (SEC)',10)
      132.             call yname ('DYNAMIC PRESSURE (DYNES/CM**2)',30)
      133.            else
      134.             call anotat(11htime, sec $,30hdynamic pressure, dynes/cm**2$,
                 1       1,1,0,0)
      135.            endif
      136.            icall=2
              c
              c     generate plots.
              c     Set up the titles and axes for the plots using DISSPLA or NCAR.
      137.   240      if (displa) then
      138.              imx=ismax(length,star(1,istat),1)
      139.              call page (10.,11.)
      140.              call area2d (6.,8.)
      141.              encode (30,280,glab) star(imx,istat),tim(imx)
      142.              call headin (glab,30,1.5,4)
      143.              encode (32,300,glab) kfile,restf
      144.              call headin (glab,32,1.5,4)
      145.              encode (48,310,glab) ksta,xstat,ystat,zstat
      146.              call headin (glab,48,1.5,4)
      147.              encode (nlabel,290,glab) label(1:nlabel)
      148.              if (darkbd) then
      149.                call swissm
      150.                call shdchr (90.,1,0.003,1)
      151.              endif
      152.              call headin (glab,nlabel,2.,1)
      153.              call disprt (tim,star(1,istat),length)
      154.              if (icall.eq.1.and.ichose.eq.3) go to 230
      155.            else
      156.              imx=ismax(length,star(1,istat),1)
      157.              encode (30,280,glab) star(imx,istat),tim(imx)
      158.              call pwrit (700,950,glab,30,1,0,0)
      159.              encode (32,300,glab) kfile,restf
      160.              call pwrit (550,30,glab,32,1,0,0)
      161.              encode (48,310,glab) ksta,xstat,ystat,zstat
      162.              call pwrit (550,50,glab,48,1,0,0)
      163.              encode (nlabel,290,glab) label(1:nlabel)
      164.              call ezxy (tim,star(1,istat),length,glab)
      165.              if (icall.eq.1.and.ichose.eq.3) go to 230
      166.            endif
      167.   250      continue
              c
              c     terminate graphics.
      168.   260         call gdone
              c
              c     close files.
      169.            close (5)
      170.            close (6)
      171.            close (7)
      172.            stop
              c
      173.   270 format (1x,a)
```

```
     281    174.   280 format ('peak ',e10.3,' at ',e10.3,'$')
     282    175.   290 format (a)
     283    176.   300 format (i2,' dumps, last dump is ',a,'$')
     284    177.   310 format ('station',i3,', x,y,z ',3e10.3)
     285    178          end
  FLIK3D        SHORT VECTOR LOOP BEGINS AT SEQ. NO.          57, P=     551b
  FLIK3D              VECTOR LOOP BEGINS AT SEQ. NO.          97, P=    1013b
  FLIK3D              VECTOR LOOP BEGINS AT SEQ. NO.         104, P=    1047a
  FLIK3D              VECTOR LOOP BEGINS AT SEQ. NO.         107, P=    1070b
```

TABLE OF STATEMENT NUMBERS (ALL ADDRESSES IN TABLES ARE IN OCTAL)

```
NUMBE

  10 UNDEF**    43L     40E
  10A530b       40L
  10BUNDEF**    40L
  20 544c       44L     42J
  30 UNDEF**    58L     57E
  30AUNDEF**    57L
  30BUNDEF**    57L
  40 601b       60L     59J
  50 UNDEF**    64L     60E
  50A610b       60L
  50B635a       60L
  60 635c       66L     59J
  70 UNDEF**    70L     66F
  70A644c       66L
  70B670d       66L
  80 671b       72L     59J
  90 UNDEF**    76L     72F
  90A700b       72L
  90B725a       72L
 100 725a       77L     71J     65J     46E
 100A551b       46L
 100B727d       46L
 110 730d       79L     37R     36E
 110A500a       36L
 110B733c       36L
 120 734a       81L     35J
 130 UNDEF**    95L     92E
 130A777a       92L
 130BUNDEF**    92L
 140 1013b      96L     94J
 150 UNDEF**    98L     97E
 150A1023a      97L
 150B1031b      97L
 160 1047a      104L    103J/2
 170 UNDEF**    105L    104F
 170A1057d      104L
 170B1066d      104L
 180 1070b      107L    103J
 190 UNDEF**    108L    107E
 190A1100b      107L
 190B1112a      107L
 200 UNDEF**    109L    101E
 200A1037a      101L
 200B1114d      101L
 210 1114d      110L    106J    87R     86E
 210A744c       86L
 210B1121b      86L
 220 1150c      120L    119J/2
 230 1174c      165J    154J    129L    119J
 240 1220b      137L    128J
```

```
   250 UNDEF**    167L     111F
   250A1124b      111L
   250B1527c      111L
   260 1527c      168L      80J
   270 FN         173L      87R      37R
   280 FN         174L     157R     141R
   290 FN         175L     163R     147R
   300 FN         176L     159R     143R
   310 FN         177L     161R     145R
 00002 473b        30L
 00003 474b        32L
 00004 546c        45L
 00005 577d        59W
 00006 1045c      103W
 00007 1142c      118L
 00008 1147a      119W
 00009 1165b      121L
 00010 1173b      124L
 00011 1211b      130L
 00012 1217b      133L
 00013 1367d      137L
 00014 1350c      148L
 00015 1524d      155L

  (SN=STATEMENT NUMBER, GSN=GENERATED STATEMENT NUMBER)
  (FN=FORMAT NUMBER, UNDEF**=UNDEFINED STATEMENT NUMBER)

TABLE OF NAMES ENCOUNTERED (ADDRESS FOR DUMMY ARGUMENT IS THE ARGUMENT NUMBER)

ADDRESS NAME     TYPE MAIN USAGE  BLOCK        SOURCE PROGRAM REFERENCES

        $CLS          EXTERNAL                  171U    170U    169U
        $EFA          EXTERNAL                  163U    159U    147U    143U
        $EFF          EXTERNAL                  163U    161U    159U    157U    147U    145U    143U    141U
        $EFI          EXTERNAL                  163U    161U    159U    157U    147U    145U    143U    141U
        $EFV          EXTERNAL                  161U/4  159U    157U/2  145U/4  143U    141U/2
        $OPN          EXTERNAL                   27U     26U     25U
        $RFA          EXTERNAL                   87U     37U
        $RFF          EXTERNAL                   87U     37U
        $RFI          EXTERNAL                   87U     37U
        $RNL          EXTERNAL                   82U     28U
        $STOP         EXTERNAL                  172U
        ADJUST        EXTERNAL                  118U
        ANOTAT        EXTERNAL                  134U    125U
        AREA2D        EXTERNAL                  140U
     26 COLOUR   *I   VARIABLE    GRFCOM         22D     20S     15D
        CONV3D        EXTERNAL                   75U     69U     63U
     34 DARKHD    I   VARIABLE    GRFCOM        148U     22D     20S     16D
        DATAIN        EXTERNAL                   73U     67U     61U
     31 DISPLA    I   VARIABLE    GRFCOM        137U    130U    121U     30U     22D     20S     15D
        DISPRF        EXTERNAL                  153U
     30 DMAX     *R   VARIABLE    GRFCOM         22D     20S
     27 DMIN     *R   VARIABLE    GRFCOM         22D     20S
    147 DSTA      R   2DIM ARRAY                129P    108S     11D
        EZXY          EXTERNAL                  164U
```

```
     FILLEC       EXTERNAL            89U    78U
 427 FLIK3D       ENTRY               10/2
 317 GDATA        NAMELIST            28P    20P
     GDONE        EXTERNAL            168U
 316 GLAB     R   VARIABLE            164P   163S   162P   161S   160P   159S   158P   157S   152P
                                      147S   146P   145S   144P   143S   142P   141S
     GPLOT        EXTERNAL            29U
     HEADIN       EXTERNAL            152U   146U   144U   142U
 407 HISTRY       NAMELIST            82P    21S
 273 I        I   VARIABLE            108U/4 107I   105U/2 104I   98U/2  97I    94U/2  92I    42U/2
                                      40I
 314 ICALL    I   VARIABLE            165U   154U   136S   127S
 267 ICHOSE   I   VARIABLE            165U   154U   119P   106U   103P   21S
 270 IF       I   VARIABLE            86I    36I
 277 II       I   VARIABLE            69P    68P    67P/2  66I
  32 IL      *I   VARIABLE  GRFCOM    54S    22D
 233 ILEF     I   1DIM ARRAY          48U    20S    17D
 255 ILINES   I   1DIM ARRAY          54U    24S    20S    19D
   2 IMAX     I   VARIABLE  GRFCOM    73P    66N    61P    49S    22D
   1 IMIN     I   VARIABLE  GRFCOM    73P    66N    61P    48S    22D
 315 IMX      I   VARIABLE            157U/2 156S   141U/2 138S
 154 INCORE   L   VARIABLE            88P    45U    38P    20S    15D
 151 INTERP   L   1DIM ARRAY          56U    20S    15D
 275 IP       I   VARIABLE            58U/2  57I
 150 IPSTAR  *I   VARIABLE            129S   120S   13P
 236 IRIG     I   1DIM ARRAY          49U    20S    18D
     ISMAX    I   EXTERNAL            156U   138U
 265 ISTA     I   VARIABLE            112U   111N   102U   101N   21S
 306 ISTAT    I   VARIABLE            164U   157U   156U   153U   141U   138U   118U   112S   108U
                                      105U   102S
 271 ISTEP    I   VARIABLE            88P    75P    74P    69P    68P    63P    62P    38P
 244 JBAK     I   1DIM ARRAY          51U    20S    18D
 241 JFNT     I   1DIM ARRAY          50U    20S    18D
 300 JJ       I   VARIABLE            75P    74P    73P    72I
  33 JL      *I   VARIABLE  GRFCOM    55S    22D
 260 JLINES   I   1DIM ARRAY          55U    24S    20S    19D
   4 JMAX     I   VARIABLE  GRFCOM    72N    51S    22D
   3 JMIN     I   VARIABLE  GRFCOM    72N    67P    61P    50S    22D
 266 JSTA     I   VARIABLE            111N   101N   21S
 304 K        I   VARIABLE            110U   108U   105U   100S
 247 KBOT     I   1DIM ARRAY          52U    20S    18D
 303 KFILE    I   VARIABLE            153U   143U   90U    90S    85S
 276 KK       I   VARIABLE            63P    62P    61P    60I
   6 KMAX     I   VARIABLE  GRFCOM    60N    53S    22D
   5 KMIN     I   VARIABLE  GRFCOM    73P    67P    60N    52S    22D
 302 KOUNT    I   VARIABLE            110S   100U   84S
 305 KSTA     I   VARIABLE            161U   145U   115U   114U   113U   112U   111I   108U/3 105U
                                      102U   101I
 301 KT       I   VARIABLE            118P   117U   116U   99U    99S    98U    83S
 252 KTOP     I   1DIM ARRAY          53U    20S    19D
 134 LABEL    CH  VARIABLE            163U   147U   94U    88P    75P    74P    69P    68P    63P
                                      62P    42U    38P    2D
 155 LADJ     L   VARIABLE            118U   21S    15D
 312 LENGTH   I   VARIABLE            164P   156P   153P   138P   118P   116S
     LIBDISP      EXTERNAL            31U
     LIBNCAR      EXTERNAL            33U
```

```
 25 LINIP     +I  VARIABLE     GRFCOM    56S    22D    15D
    LOC        I  INLINE FUNCT          129U   120U
274 LSLICE     I  VARIABLE               58U    56U    54U    53U    52U    51U    50U    49U    48U
                                         47U    46I
  0 LSTAT      I  VARIABLE     STATS     35U    20S    10D     7D
    MAIN          ENTRY
    MDMP       I  PARAMETER               6S
    MDXX       I  PARAMETER              12P   11P/3    6S
  7 MPLOT     +I  1DIM ARRAY   GRFCOM    58S    22D    19D
230 MSECT      I  1DIM ARRAY             47U    20S    17D
  0 MSLICE     I  VARIABLE     GRFCOM    59P    47S    22D
156 MSWICH     I  2DIM ARRAY             58U    20S    17D
    MXX        I  PARAMETER              9P/2   8P/3    5S
143 NAMEF     CH  VARIABLE               27P     4S     3D
  0 NBU       +I  VARIABLE     NBUR      23D    20S
264 NFILES     I  VARIABLE               86N    36N    20S
272 NLABEL     I  VARIABLE              163U   163P   152P   147U   147P    93U    93S    91S    75P
                                         74P    69P    68P    63P    62P    41U    41S    39S
    NPAR       I  PARAMETER             24S/2  19P/3  18P/4  17P/3   15P    14S
263 NSLICE     I  VARIABLE               55U    46N    20S
    NSTA       I  PARAMETER              12P   11P/2   9P/4   8P/3    5S
 11 NXX        I  VARIABLE     STATS    110U   107N   104N    99U    97N    10D
    PAGE          EXTERNAL              139U
142 PRNIF     CH  VARIABLE               26P     4S     3D
    PROF3D        EXTERNAL               74U    68U    62U
    PROGRAM       EXTERNAL                1U
  4 PRS        R  2DIM ARRAY   STATS    105U    10D     9D
146 PSTA       R  2DIM ARRAY            120P   118P   105S    11D
    PWRIT         EXTERNAL              162U   160U   158U
141 READF     CH  VARIABLE               25P     4S     3D
    READIN        EXTERNAL               45U
144 RESTF     CH  VARIABLE              159U   143U    88P    87S    75P    74P    69P    68P    63P
                                         62P    38P    37S     3D
    RESTRT        EXTERNAL               88U    38U
  1 RHS        R  2DIM ARRAY   STATS    108U    10D     8D
    SHDCHR        EXTERNAL              150U
  1 STAR       R  2DIM ARRAY   POINTEE  164P   157U   156P   153P   141U   138P    13P    12D
    SWISSM        EXTERNAL              149U
145 TIM        R  1DIM ARRAY            164P   157U   153P   141U   118P   117U    98S    11D
313 TMAX       R  VARIABLE              118P   117S
  5 TME        R  1DIM ARRAY   STATS     98U    10D     9D
  2 VLS        R  2DIM ARRAY   STATS   108U/2   10D     8D
    XNAME         EXTERNAL              131U   122U
  6 XS         R  1DIM ARRAY   STATS    113U    10D     9D
307 XSTAT      R  VARIABLE              161U   145U   113S
    YNAME         EXTERNAL              132U   123U
  7 YS         R  1DIM ARRAY   STATS    114U    10D     9D
310 YSTAT      R  VARIABLE              161U   145U   114S
 10 ZS         R  1DIM ARRAY   STATS    115U    10D     9D
311 ZSTAT      R  VARIABLE              161U   145U   115S

TABLE OF EXTERNAL NAMES
-----------------------

    $CLS          EXTERNAL              171U   170U   169U
    $FEA          EXTERNAL              163U   159U   147U   143U
```

```
$EFF       EXTERNAL     163U    161U    159U    157U    147U    145U    143U    141U
$EFI       EXTERNAL     163U    161U    159U    157U    147U    145U    143U    141U
$EFV       EXTERNAL     161U/4  159U    157U/2  145U/4  143U    141U/2
$OPN       EXTERNAL     27U     26U     25U
$REA       EXTERNAL     87U     37U
$REF       EXTERNAL     87U     37U
$REI       EXTERNAL     87U     37U
$RNL       EXTERNAL     82U     28U
$STOP      EXTERNAL     172U
ADJUST     EXTERNAL     118U
ANOTAT     EXTERNAL     134U    125U
AREA2D     EXTERNAL     140U
CONV3D     EXTERNAL     75U     69U     63U
DATAIN     EXTERNAL     73U     67U     61U
DISPRF     EXTERNAL     153U
EZXY       EXTERNAL     164U
FILEFC     EXTERNAL     89U     78U
GDONE      EXTERNAL     168U
GPLOT      EXTERNAL     29U
HEADIN     EXTERNAL     152U    146U    144U    142U
ISMAX    I EXTERNAL     156U    138U
LIBDISP    EXTERNAL     31U
LIBNCAR    EXTERNAL     33U
PAGE       EXTERNAL     139U
PROF3D     EXTERNAL     74U     68U     62U
PROGRAM    EXTERNAL     1U
PWRIT      EXTERNAL     162U    160U    158U
READIN     EXTERNAL     45U
RESTRT     EXTERNAL     88U     38U
SHDCHR     EXTERNAL     150U
SWISSM     EXTERNAL     149U
XNAME      EXTERNAL     131U    122U
YNAME      EXTERNAL     132U    123U

ABBREVIATIONS USED ABOVE (THESE ARE KEYED TO THE SOURCE LISTING LINE NUMBER)
------------- ----
A  USED IN FORTRAN  ASSIGN STATEMENT      P  USED IN CALL/FUNC CALL OR ARRAY DEF
D  DEFINED IN  DECLARATIVE STATEMENT      R  FORMAT  USED  IN A  READ  STATEMENT
E  STATEMENT NUMBER ENDING A DO LOOP      S  STORED  SO CONTENTS  MAY BE CHANGED
I  INDEX OF A DO  OR IMPLIED DO LOOP      U  NAME  USED IN EXECUTABLE  STATEMENT
J  STATEMENT NUMBER USED IN TRANSFER      W  FORMAT  USED IN A  WRITE  STATEMENT
L  SOURCE LINE OF A STATEMENT NUMBER      *  DEFINED  OR DECLARED  BUT NOT  USED
N  NAME USED AS A  DO LOOP PARAMETER      ?  TEN OR  MORE REFERENCES  TO  SYMBOL

TABLE OF PARAMETERS ENCOUNTERED
----- -- ---------- -----------

MDMP      = 1                                          MDXX      = 1
MXX       = 1                                          NPAR      = 3
NSTA      = 1
```

TABLE OF LOOPS ENCOUNTERED

LABEL	INDEX	FROM -	TO	ADDRESS -	LENGTH
110	IF	36	79	500a	234
10	I	40	43	530b	15
100	LSLICE	46	77	551b	157
30	IP	57	58		INLINE
50	KK	60	64	610b	25
70	II	66	70	644c	25
90	JJ	72	76	700b	25
210	IF	86	110	744c	155
130	I	92	95	777a	14
150	I	97	98	1023a	6
200	KSTA	101	109	1037a	56
170	I	104	105	1057d	7
190	I	107	108	1100b	12
250	KSTA	111	167	1124b	404

BLOCK NAMES AND LENGTHS IN OCTAL

1544-FLIK3D 53-#IB 517 #CL 12-STATS 35-GRFCOM 1-NBUR

STATIC SPACE (IN OCTAL)

B SAVE:	31
T SAVE:	4
CONSTANTS:	132
VARIABLES:	165
TEMPORARIES:	535
CODE:	1225
TOTAL:	2336

```
  1.          subroutine conv3d (istep,restf,label,nlabel,lplane)
        c- ----------------------------------------------------------
        c     graphics for FASI3D.
        c     Conv3d plots, at the present time, 7 distinct graphs.  The graphs to
        c  plotted have a non zero value in their positions in the array mplot.
        c  The values mplot(8) through mplot(14) refer to the 7 plots in this
        c  subroutine.
        c     The plots are, in order, a contour plot of renormalized density,
        c  a contour plot of energy/volume, a contour plot of pressure (absolute)
        c  a contour plot of mach number, a velocity vector plot, a plot of trace
        c  particles used in the calculation, and a template grid to show the real
        c  spacing of the irregular grid used in the calculations.
        c     Flik3d can handle both regularly and irregularly spaced grids.  If
        c  an irregularly spaced grid is used, the flag interp will be set,
        c  and a call will be made to the subroutine conint to interpolate onto an
        c  regular grid.
        c     Also, contour plots can be done in color.  If colour is specified,
        c  the plots will be drawn by the routine colcon.  Otherwise, they will be
        c  drawn within conv3d itself, using either DISSPLA or NCAR calls, and
        c  calling on the shorter routine discon for DISSPLA.
        c       Warning:  a large array in blank common is used by the DISSPLA
        c  routine CONMAK.  Do not use blank common for anything without allowing
        c  for that block.
        c-------------------- array dimensions ----------------------------
  2.          parameter (nnnx=72, nnny=33, nnnz=88)
  3.          parameter (nnnxp=nnnx+1, nnnyp=nnny+1, nnnzp=nnnz+1)
  4.          parameter (nnnx2=nnnx+2, nnny2=nnny+2, nnnz2=nnnz+2)
        c-------------------- plot variables ------------------------------
  5.          parameter (ndm1=72, ndm2=88)
  6.          real rvx(ndm1,ndm2), rvy(ndm1,ndm2), rvz(ndm1,ndm2)
  7           real erg(ndm1,ndm2)
  8.          real rho(ndm1,ndm2), rha(ndm1,ndm2), rhb(ndm1,ndm2)
  9.          common /pltvar/ rvx, rvy, rvz, erg, rho, rha, rhb
        c ---------- fixed arrays of hydrodynamic data --------------------
 10.          parameter (nspec=1, nsum=4*nspec)
 11.          real rcamb(nnnz2), pcamb(nnnz2), ecamb(nnnz2)
 12.          real sumpl(nsum,nnnx)
 13.          real rcmin(nnnz2), gravz(nnnz2), grapz(nnnz2), fsky(nnnz2)
        c
 14.          common /hydfix/ rcamb, pcamb, ecamb, sumpl, rcmin, gravz,
           1     grapz, sum4, shrink, fsky, ratio
        c---------------- grid arrays --------------------------------------
 15.          real xcor(nnnxp), ycor(nnny), zcor(nnnz)
 16.          real twodx(nnnxp), twody2(nnny2), twodz2(nnnz2)
 17.          real dtodx(nnnxp), dtody2(nnny2), dtodz2(nnnz2)
        c
 18.          common /grdvar/ xcor, ycor, zcor, twodx, twody2, twodz2, dtodx
           1     , dtody2, dtodz2
        c-------------------- global variables ----------------------------
 19.          common /glovar/ dtmin, dtmax, cour, time, dt, dth, rlos, nfile
           1     , nstep, lllx, llly, lllz
        c--------------------------- scratch space -------------------------
 20.          parameter (nnyz2d=nnny2*nnnz2, nvar=4*nspec)
 21           parameter (ndm12=(ndm1+2)*(ndm2+2))
 22           parameter (ndm1p2=ndm1+ndm2)
 23           real dumv(ndm12,nvar), dumy(ndm12)
```

```
 24.        common /holder/ space(ndm1,ndm2),valmin,valrng,
          +     space2(ndm1,ndm2)
 25.        common /locom1/ dumv, dumy
      c --------------------- pointers ------------------------------
 26.        real arr(ndm1,ndm2), hcor(ndm1), vcor(ndm2)
 27.        real h(ndm1p2,2), v(ndm1p2,2)
 28.        real arh(ndm1,ndm2), arv(ndm1,ndm2)
 29.        pointer (iparr,arr), (iphcor,hcor), (ipvcor,vcor)
 30.        pointer (iparh,arh), (iparv,arv)
 31.        pointer (ipminh,minh), (ipmaxh,maxh)
 32.        pointer (ipminv,minv), (ipmaxv,maxv)
      c ------------------ local declarations --------------------------
 33.        real pre(ndm1,ndm2), mac(ndm1,ndm2), gam(ndm1,ndm2)
 34.        real eint(ndm1), scr1(ndm1,ndm2), scr2(ndm1,ndm2)
 35.        equivalence (pre,dumv), (mac,dumv(1,2)), (gam,dumv(1,3))
 36.        equivalence (scr1,dumv(1,4)), (scr2,dumv(1,5)), (eint,dumy)
 37.        real rscr(ndm1), pscr(ndm1), gscr(ndm1)
 38.        real scra(ndm1), scrb(ndm1), scrc(ndm1)
 39.        equivalence (rscr,scr1), (pscr,scr1(1,2)), (gscr,scr1(1,3))
 40.        equivalence (scra,scr1(1,4)), (scrb,scr1(1,5)), (scrc,scr2)
      c --------------------------particles-------------------------------
      c   The particles are set up for two bursts only; changing the number of
      c   bursts could cause an error in data statements for ichar and imark.
 41.          parameter (nop=70,nburst=2)
 42.          parameter (npart1=3*nop*nburst*nburst)
 43.          real xp(nop,nburst), zp(nop,nburst)
 44.          integer nopp(nburst), nchk(nop,nburst)
 45.          common /part1/ xp,zp,nopp,nchk
 46.        common /nbur/ nbu
      c
 47.        integer mplot(14)
 48.        character *40 label
 49.        character *8 restf
 50.        integer ichar(nburst),imark(nburst)
 51.        real glab1(7), glab2(7), glab3(7), glab4(7)
 52.        logical lintp, colour, displa, darkhd
 53.        data ksh, ksv /30,30/
 54.        data ichar/8hx       ,8ho       /,imark/4,16/
 55.        namelist /grfdat/ mslice,imin,imax,jmin,jmax,kmin,kmax,mplot
          1     ,lintp,colour
 56.        common /grfcom/ mslice, imin, imax, jmin, jmax, kmin, kmax,
          1     mplot, lintp, colour, dmin, dmax, displa, il, jl, darkhd
 57.        common /bound/ hmin, hmax, vmin, vmax
      c
      c   first print labels.
 58.        encode (nlabel,250,glab1) label(1:nlabel)
 59.        encode (40,260,glab2) time,istep,restf
      c
      c   set window sizes.
 60.        go to (10,20,30), mslice
      c
      c   set up the pointers to specify the coordinates of the given slice
      c   X Y slice.
 61.   10   continue
 62.        ipminh loc(imin)
 63.        ipmaxh loc(imax)
```

```
      ipminv=loc(jmin)
      ipmaxv=loc(jmax)
      iphcor=loc(xcor)
      ipvcor=loc(ycor)
      iparh=loc(rvx)
      iparv=loc(rvy)
        if (displa) then
        encode (28,270,glab3) lplane,zcor(lplane)/100000.
      else
        encode (32,275,glab3) lplane,zcor(lplane)
      endif
      go to 40
c
c     Y-Z slice.
20    continue
      ipminh=loc(jmin)
      ipmaxh=loc(jmax)
      ipminv=loc(kmin)
      ipmaxv=loc(kmax)
      iphcor=loc(ycor)
      ipvcor=loc(zcor)
      iparh=loc(rvy)
      iparv=loc(rvz)
      if (displa) then
        encode (28,280,glab3) lplane,xcor(lplane)/100000.
      else
        encode (32,285,glab3) lplane,xcor(lplane)
      endif
      go to 40
c
c     X-Z slice.
30    continue
      ipminh=loc(imin)
      ipmaxh=loc(imax)
      ipminv=loc(kmin)
      ipmaxv=loc(kmax)
      iphcor=loc(xcor)
      ipvcor=loc(zcor)
      iparh=loc(rvx)
      iparv=loc(rvz)
      if (displa) then
        encode (28,290,glab3) lplane,ycor(lplane)/100000.
      else
        encode (32,295,glab3) lplane,ycor(lplane)
      endif
c
c     define plot window.
c     what section of the plane is to be drawn?
40    continue
      hmin=hcor(minh)
      hmax=hcor(maxh)
      vmin=vcor(minv)
      vmax=vcor(maxv)
      lh=maxh-minh+1
      lv=maxv-minv+1
      ni = lh / i1
```

```
113.            nj = lv / jl
114.            nij = ni + nj
115.            kh=lh
116.            kv=lv
117.            if (lintp) then
118.            delh=hmax-hmin
119.            delv=vmax-vmin
120.            kh=ndm1
121.            dhv=delh/float(kh-1)
122.            kv=1+int(delv/dhv)
123.            if (kv.le.ndm2) go to 45
124.            kv=ndm2
125.            dhv=delv/float(kv-1)
126.            kh=1+int(delh/dhv)
127.            endif
128.      45    call wsize (hmin,hmax,vmin,vmax,hlo,hhi,vlo,vhi)
         c
         c     generate contour and vector plots.
129.            ispec=1
130.            do 230 ip=8,14
131.            if (mplot(ip).eq.0) go to 230
         c     for each plot, set up the page and titles.
         c     The DISSPLA commands set the type of print, the size of the page,
         c     the size of the plot area, frame the axes and label them, write the
         c     titles, and prepare for drawing the plots.  The NCAR commands draw
         c     the titles, set the size of the subplot area, frame it and draw
         c     tickmarks.
132.      50    if (displa) then
133.              if (darkhd) then
134.                call swissm
135.                call shdchr (90.,1,0.003,1)
136.              endif
137.              call page (10.,11.)
138.              call nobrdr
139.              call area2d (8.*(hmax-hmin)/(vmax-vmin),8.)
140.              call height (.16)
141.              call headin (glab1,nlabel-1,1.6,3)
142.              call headin (glab2,25,1.6,3)
143.              call headin (glab3,27,1.6,3)
144.              call xname (' ',1)
145.              call yname (' ',1)
146.              call thkfrm (.02)
147.              call frame
148.            ystep = (vmax - vmin)/8.
149.              xstep = ystep
150.              call graf (hmin/100000.,xstep,hmax/100000.,
             +          vmin/100000.,ystep,vmax/100000.)
151.              flo=0
152.              fhi=0
153.            else
154.              call pwrit (500,1010,glab1,nlabel,1,0,0)
155.              call pwrit (350,980,glab2,40,1,0,0)
156.              call pwrit (500,950,glab3,28,1,0,0)
157.              call set (hlo,hhi,vlo,vhi,hmin,hmax,vmin,vmax,1)
158.              call perim1 (1,0,1,0)
159.              flo=0
```

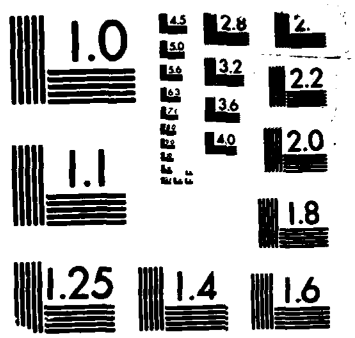
```
160.              flhi=0.
161.              endif
         c
162               go to (60,100,'10,140,170,400,420), ip-7
         c
         c     renormalized (!!!!) density contours.
         c     these plots are scaled w/respect to the ambient density.  Three
         c     species of density can be drawn (from ispec).  These are usually
         c     total density, dust density, and explosive density.  Usually only
         c     species 1 or species 1 and 2 are used.
163.        60    go to (70,80,90), ispec
         c     first, assign the pointer
164.        70    iparr=loc(space2)
165.              do 72 nv=minv,maxv
166               rscr(nv)=cvmgt(rcamb(iplane),rcamb(nv+1),mslice.eq.1)
167.              do 72 nh=minh,maxh
168.              space2(nh,nv)=rho(nh,nv)/rscr(nv)
169.        72    space2(nh,nv)=cvmgt(1.0,space2(nh,nv),space2(nh,nv).gt 0.95
              1    .and. rho(nh,nv).lt.1.05)
170.              encode (18,300,glab4)
         c     label the plot
171.              if (displa) then
172.                tleng = (hmax-hmin)/(vmax-vmin)*4.0 - xmess(glab4,17)/2.
173.                call messag (glab4,17,tleng,-.75)
174.              else
175.                call pwrit (800,980,glab4,18,1,0,0)
176.              endif
177.              go to 160
         c     first, assign the pointer (species 2)
178         80    iparr=loc(rha)
179.              encode (20,310,glab4) ispec
         c     label the plot
180.              if (displa) then
181.                tleng = (hmax-hmin)/(vmax-vmin)*4.0 - xmess(glab4,19)/2.
182.                call messag (glab4,19,tleng,-.75)
183.              else
184.                call pwrit (800,980,glab4,20,1,0,0)
185.              endif
186.              flo=dmin
187               fhi=dmax
188.              go to 160
         c     first, assign the pointer (species 3)
189.        90    iparr=loc(rhb)
190.              encode (20,310,glab) ispec
         c     label the plot
191.              if (displa) then
192                 tleng = (hmax-hmin)/(vmax-vmin)*4.0 - xmess(glab4,19)/2.
193.                call messag (glab4,19,tleng,-.75)
194.              else
195.                call pwrit (800,980,glab4,20,1,0,0)
196               endif
197.              go to 160
         c
         c     energy contours.
         c     first, assign pointer to the energy array
198.       100    iparr=loc(erg)
```

```
           encode (20,320,glab4)
c      label the plot
           if (displa) then
             tleng = (hmax-hmin)/(vmax-vmin)*4.0 - xmess(glab4,19)/2.
             call messag (glab4,19,tleng, .75)
           else
             call pwrit (800,980,glab4,20,1,0,0)
           endif
           go to 160
c
c      pressure contours.
c      pressure is calculated in the equations of state routine eos
c      first assign pointer to pressure array pre
  110      ipare=loc(pre)
           encode (23,330,glab4)
c      label the plot
           if (displa) then
             tleng = (hmax-hmin)/(vmax-vmin)*4.0 - xmess(glab4,22)/2.
             call messag (glab4,22,tleng, .75)
           else
             call pwrit (800,980,glab4,23,1,0,0)
           endif
           do 130 nv=minv,maxv
           do 120 nh=minh,maxh
  120      eint(nh)=erg(nh,nv)-0.5*(rvx(nh,nv)*rvx(nh,nv)+rvy(nh,nv)*rvy
     1     (nh,nv)+rvz(nh,nv)*rvz(nh,nv))/rho(nh,nv)
           call eos (maxh-minh+1,nspec,rho(minh,nv),eint(minh),gam(minh
     1     ,nv),pre(minh,nv),rscr(minh),pscr(minh),gscr(minh),scra(minh)
     2     ,scrb(minh),scrc(minh),rha(minh,nv),rhb(minh,nv))
  130      continue
           go to 160
c
c      mach number contours.
c      first assign pointer to mach number array mac
  140      ipare=loc(mac)
           encode (13,340,glab4)
c      label the plot
           if (displa) then
             tleng = (hmax-hmin)/(vmax-vmin)*4.0 - xmess(glab4,11)/2.
             call messag (glab4,11,tleng,-.75)
           else
             call pwrit (800,980,glab4,13,1,0,0)
           endif
c      calculate pressure from eos to use in calculating mach number
           do 146 nv=minv,maxv
           do 145 nh=minh,maxh
  145      eint(nh)=erg(nh,nv)-0.5*(rvx(nh,nv)*rvx(nh,nv)+rvy(nh,nv)*rvy
     1     (nh,nv)+rvz(nh,nv)*rvz(nh,nv))/rho(nh,nv)
           call eos (maxh-minh+1,nspec,rho(minh,nv),eint(minh),gam(minh
     1     ,nv),pre(minh,nv),rscr(minh),pscr(minh),gscr(minh),scra(minh)
     2     ,scrb(minh),scrc(minh),rha(minh,nv),rhb(minh,nv))

           [illegible] minv,maxv
           [illegible] minh,maxh
           [illegible]((rvx(nh,nv)*rvx(nh,nv)+rvy(nh,nv)*rvy(nh,nv)
     1     [illegible](rho(nh,nv)*gam(nh,nv)*pre(nh,nv)))
```

AD-A193 152 NUMERRCAL MODELING OF AIRBLAST(U) SCIENCE APPLICATIONS 3/3
INTERNATIONAL CORP MCLEAN VA M A FRY JUN 87
SAIC-87/1701 N00014-86-C-2197

UNCLASSIFIED F/G 19/11 NN



MICROCOPY RESOLUTION TEST CHART
URFAU OF STANDARDS-1963-A

```
237.           go to 160
238.     160   continue
         c
         c     plot contours.
         c     this section actually draws the contour lines.  If lintp is true, the
         c     irregular grid of values is interpolated onto a regular grid.  If
         c     colour is true, the routine colcon draws contours in color.
         c     The DISSPLA contours are actually draw by the subroutine discon, the
         c     NCAR contours by the routine conrec.
239.           if (displa) then
240.             if (lintp) then
241.               call conint (arr(minh,minv),hcor(minh),vcor(minv),ndm1,lh,
              1      lv,scr1,kh,kv)
242.               if (colour) then
243.                 call colcon (scr1,1,1,kh,kv)
244.               else
245.                 call discon (scr1,ndm1,space,kh,kv)
246.               endif
247.             else
248.               if (colour) then
249.                 call colcon (arr,minh,minv,kh,kv)
250.               else
251.                 call discon (arr(minh,minv),ndm1,space,kh,kv)
252.               endif
253.             endif
254.           else
255.             if (lintp) then
256.               call conint (arr(minh,minv),hcor(minh),vcor(minv),ndm1,lh,
              1      lv,scr1,kh,kv)
257.               if (colour) then
258.                 call colcon (scr1,1,1,kh,kv)
259.               else
260.                 call conrec (scr1,ndm1,kh,kv,flo,fhi,0,-1,0,0)
261.               endif
262.             else
263.               if (colour) then
264.                 call colcon (arr,minh,minv,kh,kv)
265.               else
266.                 call conrec(arr(minh,minv),ndm1,kh,kv,flo,fhi,0,-1,0,0)
267.               endif
268.             endif
269.             call framen
270.           endif
         c
         c     go back and draw the other species of density, if any
271.           ispec=ispec+1
272.           if (ispec.le.nspec.and.ip-7.eq.1) go to 50
273.           go to 230
         c
         c     plot velocity vectors.
         c     the velocity vector plots are plotted using the NCAR routine velvct,
         c     from information in arrays of h and v velocity components.
274.     170   encode (18,350,glab4)
         c     label the plot
275.           if (displa) then
276.             fleng   (hmax hmin)/(vmax vmin)*4.0   xmess(glab4,17)/2
```

```
277.            call messag (glab4,17,tleng,-.75)
278.          else
279.            call pwrit (800,980,glab4,18,1,0,0)
280.          endif
      c
      c     first calculate velocities from momenta.
281.          do 180 nv=minv,maxv
282.          do 180 nh=minh,maxh
283.          pre(nh,nv)=1.0/rho(nh,nv)
284.          arh(nh,nv)=arh(nh,nv)*pre(nh,nv)
285.   180    arv(nh,nv)=arv(nh,nv)*pre(nh,nv)
      c
      c     plot vectors. interpolate onto uniform grid if fintp=.true.
286.          velmax=0.0
287.          velmin=1.e50
288.          if (fintp) then
289.          call conint (arh(minh,minv),hcor(minh),vcor(minv),ndm1,lh,lv
         1    ,scr1,ksh,ksv)
290.          call conint (arv(minh,minv),hcor(minh),vcor(minv),ndm1,lh,lv
         1    ,scr2,ksh,ksv)
291.          do 200 nv=1,ksv
292.          do 190 nh=1,ksh
293.   190    pre(nh,nv)=scr1(nh,nv)*scr1(nh,nv)+scr2(nh,nv)*scr2(nh,nv)
294           velmax=amax1(velmax,pre(ismax(ksh,pre(1,nv),1),nv))
295           velmin=amin1(velmin,pre(ismin(ksh,pre(1,nv),1),nv))
296.   200    continue
297.          velmax=sqrt(velmax)
298           velmin=sqrt(velmin)
299.          velmin=velmin+0.05*(velmax-velmin)
      c     draw the velocity vectors.  Only NCAR is made use of (scaled to fit
      c     the DISSPLA axes, if necessary), as only NCAR has a velocity plotting
      c     routine.  libncar is called to initialize NCAR graphics, set to scale
      c     the plot, if displa=.true.
300.          if (displa) then
301.            call libncar
302.            xa=(1.0 - .8*(hmax-hmin)/(vmax-vmin))/2. + 0.025
303.            xb=1.05 - xa
304.            call set (xa,xb,.110,.838,hmin,hmax,vmin,vmax,1)
305.            call velvct (scr1,ndm1,scr2,ndm1,ksh,ksv,velmin,velmax,-1,
         1      0,0,0,0)
306.          else
307.            call velvct (scr1,ndm1,scr2,ndm1,ksh,ksv,velmin,velmax,-1,
         1      0,0,0,0)
308.          endif
309           else
310.          do 220 nv=minv,minv+kv-1
311.          do 210 nh=minh,minh+kh-1
312.   210    pre(nh,nv)=arh(nh,nv)*arh(nh,nv)+arv(nh,nv)*arv(nh,nv)
313.          velmax=amax1(velmax,pre(ismax(kh,pre(minh,nv),1)+minh-1,nv))
314.          velmin=amin1(velmin,pre(ismin(kh,pre(minh,nv),1)+minh-1,nv))
315.   220    continue
316.          velmax=sqrt(velmax)
317.          velmin=sqrt(velmin)
318.          velmin=velmin+0.05*(velmax-velmin)
      c     plot the vector plot (as described above), using uninterpolated grid.
319.          if (displa) then
```

```
320.              call libncar
321.              xa=(1.0 - .8*(hmax-hmin)/(vmax-vmin))/2. + .025
322.              xb=1.05 - xa
323.              call set (xa,xb,.110,.838,hmin,hmax,vmin,vmax,1)
324.              call velvct (arh(minh,minv),ndm1,arv(minh,minv),ndm1,kh,kv
          1    ,velmin,velmax,1,0,0,0.0)
325.            else
326.              call velvct (arh(minh,minv),ndm1,arv(minh,minv),ndm1,kh,kv
          1    ,velmin,velmax,-1,0,0,0.0)
327.            endif
328.            endif
329.            call framen
330.            if (displa) call endpl (0)
331.            go to 230
332.            return
     c
     c     particles
     c
333.  400       continue
334.            encode (11,360,glab4)
     c     using either NCAR or DISSPLA, the plot is labeled, and markers are
     c     placed at points showing the positions of the trace particles.  A
     c     different marker is used for each burst, from the arrays ichar or
     c     imark, respectively.
335.            if (displa) then
336.              tleng = (hmax-hmin)/(vmax-vmin)*4.0 - xmess(glab4,10)/2.
337.              call messag (glab4,10,tleng,-0.75)
338.              do 405 k=1,nbu
339.              call marker (imark(k))
340.              do 406 ijk = 1, nopp(k)
341.              xp(ijk,k) = 0.00001 * xp(ijk,k)
342.              zp(ijk,k) = 0.00001 * zp(ijk,k)
343.   406        continue
344.              call curve (xp(1,k),zp(1,k),nopp(k),-1)
345.   405        continue
346.              call endpl(0)
347.            else
348.              call pwrit (800,980,glab4,11,1,0,0)
349.              write (6,1021)
350.              do 410 k=1,nbu
351.              do 410 i=1,nopp(k)
352.              call points(xp(i,k),zp(i,k),1,ichar(k),0)
353.              write (6,1022) i,xp(i,k),zp(i,k)
354.  410         continue
355.              call framen
356.            endif
357.            go to 230
     c
     c          grid
     c     this draws the irregular grid on which the original calculation was
     c     done.  this serves to show where the resolution was finest, and where
     c     the most interpolation was done, on the contour plots.
     c
358.   420      continue
     c
359.            do 421 i = 1, ni
```

```
360.            h(i,1) = hcor(minh + il + i - 1)
361.            h(i,2) = h(i,1)
362.   421      continue
          c
363.            do 422 j = 1, nj
364.            jj = j + ni
365.            v(jj,1) = vcor(minv + jl + j - 1)
366.            v(jj,2) = v(jj,1)
367.   422      continue
          c
368.            do 423 i = 1, ni
369.            v(i,1) = v(ni+1,1)
370.            v(i,2) = v(nij,1)
371.   423      continue
          c
372.            do 424 j = ni+1, nij
373.            h(j,1) = h(1,1)
374.            h(j,2) = h(ni,1)
375.   424      continue
          c
376.            if( nij .lt. ndmlp2 ) then
377.              do 425 j = nij + 1, ndmlp2
378.              v(j,1) = v(nij,1)
379.              v(j,2) = v(nij,2)
380.              h(j,1) = h(nij,1)
381.              h(j,2) = h(nij,2)
382.   425        continue
383.            endif
          c
          c     draw the grid itself
384.            if (displa) then
385.              do 426 j=1,ndmlp2
386.              space(1,1)=h(j,1)/100000.
387.              space(2,1)=h(j,2)/100000.
388               space(1,2)=v(j,1)/100000.
389.              space(2,2)=v(j,2)/100000.
390.              call curve (space(1,1),space(1,2),2,0)
391.   426        continue
392.              call endpl (0)
393.            else
394.              call anotat (th$,th$,1,1,0,0)
395.              call agseti( ' win  ', 1 )
396.              call agsetf( ' grid/left. ', hlo )
397.              call agsetf( ' grid/right. ', hhi )
398.              call agsetf( ' grid/bottom. ',vlo )
399.              call agsetf( ' grid/top. ', vhi )
400.              call agsetf( ' x/minimum. ', h(1,1) )
401.              call agsetf( ' x/maximum. ', h(ni,1) )
402.              call agsetf( ' y/minimum. ', v(ni+1,1) )
403.              call agsetf( ' y/maximum. ', v(nij,1) )
404.              call agseti( ' x/nice. ', 0. )
405.              call agsetf( ' y/nice. ', 0. )
406.              call agseti( ' row. ', 2. )
407.              call ezmxy( h, v., ndmlp2, ndmlp2, 2. 0 )
408.            endif
          c
```

```
409.  230      continue
410.           return
          c
          c     this entry is a fossilized routine, implemented but not used in the
          c     current version of the code.  Ignore it, or better still, rip it out,
          c     if you have the time and inclination.
411.           entry grfin(lpl)
          c ----------------------------------------------------------------------
          c     reads in data for in-line graphics.
          c ----------------------------------------------------------------------
412.           read (5,grfdat)
413.           write (6,grfdat)
414.           if (mslice.eq.1) lplane=kmin
415.           if (mslice.eq.2) lplane=imin
416.           if (mslice.eq.3) lplane=jmin
417.           lpl=lplane
418.           do 240 nv=1,ndm2
419.           do 240 nh=1,ndm1
420.           rho(nh,nv)=cvmgt(rcamb(lpl+1),rcamb(nv+1),mslice.eq.1)
421.           rvx(nh,nv)=0.0
422.           rvy(nh,nv)=0.0
423.           rvz(nh,nv)=0.0
424.   240     erg(nh,nv)=cvmgt(ecamb(lpl+1),ecamb(nv+1),mslice.eq.1)
425.           call qplot (lnu,4hplot,12)
426.           if (displa) then
427.             call libdisp
428.           else
429.             call libncar
430.           endif
431.           return
          c
432.           entry grfend
          c ----------------------------------------------------------------------
          c     terminates NCAR or DISSPLA for inline graphics.
          c ----------------------------------------------------------------------
433.           call qdone
434.           return
          c
          c     formats, and other important goodies, for the titles and labels.
435.    250 format (a)
436.    260 format ('time=',f5.1,' sec, step',i5,', dump ',a)
437.    270 format ('X Y plane ',i3,' at z=',f5.1,' km$')
438.    275 format ('X-Y plane ',i3,' at z=',e9.3,' cm$')
439.    280 format ('Y Z plane ',i3,' at x=',f5.1,' km$')
440.    285 format ('Y Z plane ',i3,' at x=',e9.3,' cm$')
441.    290 format ('X Z plane ',i3,' at y=',f5.1,' km$')
442.    295 format ('X Z plane ',i3,' at y=',e9.3,' cm$')
443.    300 format ('density (g/cm**3)$')
444.    310 format ('density ',i1,' (g/cm**3)$')
445.    320 format ('energy (ergs/cm**3)$')
446.    330 format ('pressure (dynes/cm**2)$')
447.    340 format ('mach number$')
448.    350 format ('velocity (cm/sec)$')
449.    360 format ('particles $')
450.   1021 format (/1x,'part       zp             zp')
451.   1022 format (1x,i3,2(2x,1pe14.7))
```

```
  902     452.         end
 CONV3D               VECTOR LOOP BEGINS AT SEQ. NO.         167, P=     2374a
 CONV3D               VECTOR LOOP BEGINS AT SEQ. NO.         216, P=     2677b
 CONV3D               VECTOR LOOP BEGINS AT SEQ. NO.         230, P=     3035c
 CONV3D               VECTOR LOOP BEGINS AT SEQ. NO.         235, P=     3134a
 CONV3D               VECTOR LOOP BEGINS AT SEQ. NO.         282, P=     3423a
 CONV3D               VECTOR LOOP BEGINS AT SEQ. NO.         292, P=     3526b
 CONV3D               VECTOR LOOP BEGINS AT SEQ. NO.         311, P=     3661b
 CONV3D               VECTOR LOOP BEGINS AT SEQ. NO.         340, P=     4076b
0   AT SEQUENCE NUMBER -    352.
 PRNAME CONV3D   COMMENT -     DEPENDENCY INVOLVING ARRAY "XP" IN SEQUENCE NUMBER 353      ****************************** P=02506300
 EXPLANATION:  ARRAY USED AS AN ARGUMENT TO A SUBROUTINE/FUNCTION
0   AT SEQUENCE NUMBER -    352.
 PRNAME CONV3D   COMMENT -     DEPENDENCY INVOLVING ARRAY "ZP" IN SEQUENCE NUMBER 353      ****************************** P=02506300
 EXPLANATION:  ARRAY USED AS AN ARGUMENT TO A SUBROUTINE/FUNCTION
 CONV3D               VECTOR LOOP BEGINS AT SEQ. NO.         359, P=     4220a
 CONV3D               VECTOR LOOP BEGINS AT SEQ. NO.         363, P=     4241d
0   AT SEQUENCE NUMBER -    369.
 PRNAME CONV3D   COMMENT -     DEPENDENCY INVOLVING ARRAY "V" ***************************************************** P=0250024C
 EXPLANATION:  AMBIGUOUS OR CONFLICTING SUBSCRIPTS
0   AT SEQUENCE NUMBER -    369.
 PRNAME CONV3D   COMMENT -     DEPENDENCY INVOLVING ARRAY "V" IN SEQUENCE NUMBER 370       ****************************** P=02506300
 EXPLANATION:  AMBIGUOUS OR CONFLICTING SUBSCRIPTS
 CONV3D               VECTOR LOOP BEGINS AT SEQ. NO.         368, P=     4266a
 CONV3D  CONDITIONAL VECTOR LOOP BEGINS AT SEQ. NO.         368, P=     4266a
0   AT SEQUENCE NUMBER -    373.
 PRNAME CONV3D   COMMENT -     DEPENDENCY INVOLVING ARRAY "H" ***************************************************** P=0250024C
 EXPLANATION:  AMBIGUOUS OR CONFLICTING SUBSCRIPTS
0   AT SEQUENCE NUMBER -    373.
 PRNAME CONV3D   COMMENT -     DEPENDENCY INVOLVING ARRAY "H" IN SEQUENCE NUMBER 374       ****************************** P=02506300
 EXPLANATION:  AMBIGUOUS OR CONFLICTING SUBSCRIPTS
0   AT SEQUENCE NUMBER -    378.
 PRNAME CONV3D   COMMENT -     DEPENDENCY INVOLVING ARRAY "V" ***************************************************** P=0250024C
 EXPLANATION:  AMBIGUOUS OR CONFLICTING SUBSCRIPTS
0   AT SEQUENCE NUMBER -    379.
 PRNAME CONV3D   COMMENT -     DEPENDENCY INVOLVING ARRAY "V" ***************************************************** P=0250024C
 EXPLANATION:  AMBIGUOUS OR CONFLICTING SUBSCRIPTS
0   AT SEQUENCE NUMBER -    380.
 PRNAME CONV3D   COMMENT -     DEPENDENCY INVOLVING ARRAY "H" ***************************************************** P=0250024C
 EXPLANATION:  AMBIGUOUS OR CONFLICTING SUBSCRIPTS
 CONV3D               VECTOR LOOP BEGINS AT SEQ. NO.         377, P=     4340b
 CONV3D  CONDITIONAL VECTOR LOOP BEGINS AT SEQ. NO.         377, P=     4340b
0   AT SEQUENCE NUMBER -    390.
 PRNAME CONV3D   COMMENT -     DEPENDENCY INVOLVING ARRAY "SPACE" IN SEQUENCE NUMBER 386   ****************************** P=0250305C
 EXPLANATION:  ARRAY USED AS AN ARGUMENT TO A SUBROUTINE/FUNCTION
```

```
0   AT SEQUENCE NUMBER -    390.
 PRNAME CONV3D   COMMENT -      DEPENDENCY INVOLVING ARRAY "SPACE" IN SEQUENCE NUMBER 387      *************************** P=0250305C
 EXPLANATION:  ARRAY USED AS AN ARGUMENT TO A SUBROUTINE/FUNCTION
0   AT SEQUENCE NUMBER -    390.
 PRNAME CONV3D   COMMENT -      DEPENDENCY INVOLVING ARRAY "SPACE" IN SEQUENCE NUMBER 388      *************************** P=0250305C
 EXPLANATION:  ARRAY USED AS AN ARGUMENT TO A SUBROUTINE/FUNCTION
 CONV3D              VECTOR LOOP BEGINS AT SEQ. NO.          419, P-     4575b
```

```
 TABLE OF STATEMENT NUMBERS (ALL ADDRESSES IN TABLES ARE IN OCTAL)

  NUMBER USE    SOURCE PROGRAM REFERENCES

     10 1727c      61L     60J
     20 1775c      76L     60J
     30 2043c      91L     60J
     40 2111a     105L     90J     75J
     45 2162b     128L    123J
     50 2177b     272J    132L
     60 2360b     163L    162J
     70 2366b     164L    163J
     72 UNDEF**   169L    167E    165E
     72A2374a     165L
     72B2431d     165L
     72C2415b     167L
     72D2427a     167L
     80 2467d     178L    163J
     90 2533a     189L    163J
    100 2574b     198L    162J
    110 2634a     207L    162J
    120 UNDEF**   217L    216E
    120A2707d     216L
    120B2731a     216L
    130 UNDEF**   219L    215E
    130A2677b     215L
    130B2771d     215L
    140 2772b     221L    162J
    145 UNDEF**   231L    230E
    145A3046a     230L
    145B3067b     230L
    146 UNDEF**   233L    229E
    146A3035c     229L
    146B3130a     229L
    150 UNDEF**   236L    235E    234E
    150A3134a     234L
    150B3174a     234L
    150C3144c     235L
    150D3171b     235L
    160 3174c     238L    237J    220J    206J    197J    188J    177J
    170 3361c     274L    162J
    180 UNDEF**   285L    282E    281E
    180A3423a     281L
    180B3455b     281L
    180C3436c     282L
    180D3452c     282L
    190 UNDEF**   293L    292E
    190A3535c     292L
    190B3546d     292L
    200 UNDEF**   296L    291E
    200A3526b     291L
    200B3572a     291L
    210 UNDEF**   312L    311E
    210A3675c     311L
```

210B3705d	311I					
220 UNDEF**	315I	310F				
220A3661b	310L					
220B3736c	310I					
230 4533b	409I	357J	331J	273J	131J	130E
230A2175d	130L					
230BUNDEF**	130I					
240 UNDEF**	424L	419E	418E			
240A4575b	418L					
240BUNDEF**	418L					
240C4611b	419L					
240DUNDEF**	419L					
250 FN	435L	58R				
260 FN	436L	59R				
270 FN	437L	71R				
275 FN	438L	73R				
280 FN	439L	86R				
285 FN	440L	88R				
290 FN	441L	101R				
295 FN	442L	103R				
300 FN	443L	170R				
310 FN	444I	190R	179R			
320 FN	445L	199R				
330 FN	446L	208R				
340 FN	447L	222R				
350 FN	448L	274R				
360 FN	449L	334R				
400 4046a	333L	162J				
405 UNDEF**	345L	338E				
405A4076b	338L					
405B4136b	338L					
406 UNDEF**	343I	340F				
406A4113b	340L					
406B4124b	340L					
410 UNDEF**	354L	351E	350E			
410A4160a	350L					
410B4216c	350L					
410C4165b	351L					
410D4213d	351L					
420 4220a	358L	162J				
421 UNDEF**	362L	359E				
421A4232a	359L					
421B4241d	359I					
422 UNDEF**	367I	363E				
422A4253d	363I					
422B4266a	363I					
423 UNDEF**	371L	368E				
423A4302d	368L					
423B4324a	368L					
424 UNDEF**	375L	372E				
424A4332d	372L					
424B4340b	372L					
425 UNDEF**	382I	377F				
425A4355c	377L					
425B4405a	377I					
426 UNDEF**	391L	385F				

426A4411c		385I	
426BUNDLE**		385L	
1021	FN	450I	349W
1022	FN	451L	353W
00002	1726a	60W	
00003	1762d	70L	
00004	1775a	72I	
00005	2030d	85I	
00006	2043a	87I	
00007	2076d	100I	
00008	2111a	102L	
00009	2162b	117I	
00010	2302b	132L	
00011	2206d	133L	
00012	2350a	153L	
00013	2354d	162W	
00014	2364d	163W	
00015	2457d	171L	
00016	2467b	174I	
00017	2521a	180I	
00018	2530c	183I	
00019	2564b	191L	
00020	2573d	194I	
00021	2624a	200L	
00022	2633c	203I	
00023	2663d	209L	
00024	2673b	212L	
00025	3022a	223L	
00026	3031c	226I	
00027	3260c	239I	
00028	3236b	240L	
00029	3227d	242L	
00030	3235d	244L	
00031	3260a	247L	
00032	3245d	248L	
00033	3260a	250I	
00034	3354a	254L	
00035	3325b	255L	
00036	3312d	257L	
00037	3324d	259L	
00038	3353a	262L	
00039	3334d	263I	
00040	3353a	265I	
00041	3407c	275L	
00042	3417a	278L	
00043	3654c	288I	
00044	3640c	300L	
00045	3654a	306I	
00046	4037a	309I	
00047	4014b	319L	
00048	4037a	325I	
00049	4043a	330I	
00050	4140d	335I	
00051	4217c	347L	
00052	4405a	376L	
00053	4432b	384I	

```
00054 4533b     393L
00055 4563b     414I
00056 4566c     415I
00057 4571d     416I
00058 4637c     426I
00059 4640c     428L
00060 4313b     371E
00061 4324a     371E
00062 UNDEF**   371E
00063 4316b     371E
00064 4371d     382E
00065 4405a     382E
00066 UNDEF**   382E
00067 4376a     382E
 (SN=STATEMENT NUMBER, GSN=GENERATED STATEMENT NUMBER)
 (FN=FORMAT NUMBER, UNDEF*=UNDEFINED STATEMENT NUMBER)
TABLE OF NAMES ENCOUNTERED (ADDRESS FOR DUMMY ARGUMENT IS THE ARGUMENT NUMBER)
--------------------------

ADDRESS NAME    TYPE MAIN USAGE  BLOCK      SOURCE PROGRAM REFERENCES

        $EFA         EXTERNAL                 59U    58U
        $EFF         EXTERNAL                334U   274U   222U   208U   199U   190U   179U   170U   103U
                                             101U    88U    86U    73U    71U    59U    58U
        $EFI         EXTERNAL                334U   274U   222U   208U   199U   190U   179U   170U   103U
                                             101U    88U    86U    73U    71U    59U    58U
        $EFV         EXTERNAL                190U   179U  103U/2 101U/2  88U/2  86U/2  73U/2  71U/2  59U/2
        $RNL         EXTERNAL                412U
        $WFF         EXTERNAL                353U   349U
        $WFI         EXTERNAL                353U   349U
        $WFV         EXTERNAL                353U/3
        $WNL         EXTERNAL                413U
        AGSETF       EXTERNAL                406U   405U   404U   403U   402U   401U   400U   399U   398U
                                             397U   396U
        AGSETI       EXTERNAL                395U
        AMAX1    R   INLINE FUNCT.           313U   294U
        AMIN1    R   INLINE FUNCT.           314U   295U
        ANDTAT       EXTERNAL                394U
        AREA2D       EXTERNAL                139U
      4 ARH      R   2DIM ARRAY  POINTEE     326P   324P  312U/2  289P   284U   284S    30P    28D
      1 ARR      R   2DIM ARRAY  POINTEE     266P   264P   256P   251P   249P   241P    29P    26D
      5 ARV      R   2DIM ARRAY  POINTEE     326P   324P  312U/2  290P   285U   285S    30P    28D
        COLCON       EXTERNAL                264U   258U   249U   243U
     26 COLOUR   I   VARIABLE    GRFCOM      263U   257U   248U   242U    56D    55S    52D
        CONINI       EXTERNAL                290U   289U   256U   241U
        CONREC       EXTERNAL                266U   260U
   1645 CONV3D       ENTRY                   432D   411D   1D/2
        CURVE        EXTERNAL                390U   344U
        CVMGT        INLINE FUNCT.           424U   420U   169U   166U
     34 DARKHD   L   VARIABLE    GRFCOM      133U    56D    52D
   1560 DELH     R   VARIABLE                126P   121U   118S
   1561 DELV     R   VARIABLE                125U   122P   119S
   1562 DHV      R   VARIABLE                126U   125S   122U   121S
        DISCON       EXTERNAL                251U   245U
```

```
     31 DISPLA    I   VARIABLE     GRFCOM        426U    384U    335U    330U    319U    300U    275U    239U    223U
                                                 209U    200U    191U    190U    171U    132U    100U     85U     70U
                                                  56D     52D
     30 DMAX      R   VARIABLE     GRFCOM        187U     56D
     27 DMIN      R   VARIABLE     GRFCOM        186U     56D
    264 FCAMB     R   1DIM ARRAY   HYDFIX        424P/2   14D     11D
 101024 FINI      R   1D EQ  ARRAY LOCOM1        232P    231S    218P    217S     36D     34D
        ENDPL         EXTERNAL                   392U    346U    330U
        EOS           EXTERNAL                   232U    218U
  45100 FRG       R   2DIM ARRAY   PLTVAR        424S    231U    217U    198P      9D      7D
        EZMXY         EXTERNAL                   407U
   1574 FHI       R   VARIABLE                   266P    260P    187S    160S    152S
   1573 FLO       R   VARIABLE                   266P    260P    186S    159S    151S
        FLOAT     R   INLINE FUNCT.              125U    121U
        FRAME         EXTERNAL                   147U
        FRAMEN        EXTERNAL                   355U    329U    269U
  32010 GAM       R   2D EQ. ARRAY LOCOM1        236U    232P    218P     35D     33D
        GDONE         EXTERNAL                   433U
   1600 GLAB      R   VARIABLE                   190S
   1513 GLAB1     R   1DIM ARRAY                 154P    141P     58S     51D
   1522 GLAB2     R   1DIM ARRAY                 155P    142P     59S     51D
   1531 GLAB3     R   1DIM ARRAY                 156P    143P    103S    101S     88S     86S     73S     71S     51D
   1540 GLAB4     R   1DIM ARRAY                 348P    337P    336P    334S    279P    277P    276P    274S    227P
                                                 225P    224P    222S    213P    211P    210P    208S    204P    202P
                                                 201P    199S    195P    193P    192P    184P    182P    181P    179S
                                                 175P    173P    172P    170S     51D
        GPLOT         EXTERNAL                   425U
        GRAF          EXTERNAL                   150U
   1612 GRFDAT        NAMELIST                   413P    412P     55S
   4644 GRFEND        ENTRY                      432D/2
   4542 GRFIN         ENTRY                      411D/2
  47234 GSCR      R   1D EQ. ARRAY LOCOM1        232P    218P     39D     37D
    276 H         R   2DIM ARRAY                 407P    401P    400P    387U    386U    381U    381S    380U    380S
                                                 374U    374S    373U    373S    361U    361S    360S     27D
      2 HCOR      R   1DIM ARRAY   POINTEE       360U    290P    289P    256P    241P    107U    106U     29P     26D
        HEADIN        EXTERNAL                   143U    142U    141U
        HEIGHT        EXTERNAL                   140U
   1564 HHI       R   VARIABLE                   397P    157P    128P
   1563 HLO       R   VARIABLE                   396P    157P    128P
      1 HMAX      R   VARIABLE     BOUND         336U    323P    321U    304P    302U    276U    224U    210U    201U
                                                 192U    181U    172U    157P    150P    139U    128P    118U    107S
                                                  57D
      0 HMIN      R   VARIABLE     BOUND         336U    323P    321U    304P    302U    276U    224U    210U    201U
                                                 192U    181U    172U    157P    150P    139U    128P    118U    106S
                                                  57D
   1607 I         I   VARIABLE                   370U    369U    368I    361U/2  360U/2  359I    353U/3  352U/2  351I
   1507 ICHAR     I   1DIM ARRAY                 352P     54S     50D
   1606 IJK       I   VARIABLE                   342U/2  341U/2  340I
     32 IL        I   VARIABLE     GRFCOM        360U    112U     56D
   1511 IMARK     I   1DIM ARRAY                 339P     54S     50D
      2 IMAX      I   VARIABLE     GRFCOM         93P     63P     56D     55S
      1 IMIN      I   VARIABLE     GRFCOM        415U     92P     62P     56D     55S
        INT       I   INLINE FUNCT.              126U    122U
   1570 IP        I   VARIABLE                   272U    162P    131U    130I
   1501 IPARH    *I   VARIABLE                    98S     83S     68S     30P
   1476 IPARR    *I   VARIABLE                   221S    207S    198S    189S    178S    164S     29P
```

```
 1502 IPARV    *I  VARIABLE                  99S    84S    69S    30P
 1477 IPHCOR   *I  VARIABLE                  96S    81S    66S    29P
 1504 IPMAXI   *I  VARIABLE                  93S    78S    63S    31P
 1506 IPMAXV   *I  VARIABLE                  95S    80S    65S    32P
 1503 IPMINI   *I  VARIABLE                  92S    77S    62S    31P
 1505 IPMINV   *I  VARIABLE                  94S    79S    64S    32P
 1500 IPVCOR   *I  VARIABLE                  97S    82S    67S    29P
      ISMAX     I  EXTERNAL                 313U   294U
      ISMIN     I  EXTERNAL                 314U   295U
 1567 ISPEC     I  VARIABLE                 272U   271U   271S   190U   179U   163P   129S
    1 ISTEP     I  VARIABLE     DUM.ARG.     59U     1D
 1610 J         I  VARIABLE                 389U   388U   387U   386U   385I   381U   380U   379U   378U
                                            377I   374U   373U   372I   365U   364U   363I
 1611 JJ        I  VARIABLE                 366U/2 365U   364S
   33 JL        I  VARIABLE     GRFCOM      365U   113U    56D
    4 JMAX      I  VARIABLE     GRFCOM       78P    65P    56D    55S
    3 JMIN      I  VARIABLE     GRFCOM      416U    77P    64P    56D    55S
 1605 K         I  VARIABLE                 353U/2 352U/3 351N   350I   344U/3 342U/2 341U/2 340N   339U
                                            338I
 1556 KH        I  VARIABLE                 326P   324P   314P   313P   311N   266P   264P   260P   258P
                                            256P   251P   249P   245P   243P   241P   126S   121P   120S
                                            115S
    6 KMAX      I  VARIABLE     GRFCOM       95P    80P    56D    55S
    5 KMIN      I  VARIABLE     GRFCOM      414U    94P    79P    56D    55S
 1547 KSH       I  VARIABLE                 307P   305P   295P   294P   292N   290P   289P    53S
 1550 KSV       I  VARIABLE                 307P   305P   291N   290P   289P    53S
 1557 KV        I  VARIABLE                 326P   324P   310N   266P   264P   260P   258P   256P   251P
                                            249P   245P   243P   241P   125P   124S   123U   122S   116S
    3 LABEL    CH  VARIABLE     DUM.ARG.     58U    48D     1D
 1551 LH        I  VARIABLE                 290P   289P   256P   241P   115U   112U   110S
      LIBDISP      EXTERNAL                 427U
      LIBNCAR      EXTERNAL                 429U   320U   301U
   25 LINTP     L  VARIABLE     GRFCOM      288U   255U   240U   117U    56D    55S    52D
      LOC       I  INLINE FUNCT.            221U   207U   198U   189U   178U   164U    99U    98U    97U
                                             96U    95U    94U    93U    92U    84U    83U    82U    81U
                                             80U    79U    78U    77U    69U    68U    67U    66U    65U
                                             64U    63U    62U
    6 LPI       I  VARIABLE     DUM.ARG.    424U   420U   417S   411D
    5 LPLANE    I  VARIABLE     DUM.ARG.    417U   416S   415S   414S   166U   103U/2 101U/2  88U/2  86U/2
                                             73U/2  71U/2   1D
 1552 LV        I  VARIABLE                 290P   289P   256P   241P   116U   113U   111S
15004 MAC       R  2D EQ  ARRAY LOCOM1      236S   221P    35D    33D
      MAIN.        ENTRY
      MARKER       EXTERNAL                 339U
    7 MAXH      I  VARIABLE     POINTEE     282N   235N   232P   230N   218P   216N   167N   110U   107U
                                             31P
   11 MAXV      I  VARIABLE     POINTEE     281N   234N   229N   215N   165N   111U   109U    32P
      MESSAG       EXTERNAL                 337U   277U   225U   211U   202U   193U   182U   173U
    6 MINH      I  VARIABLE     POINTEE     360U   326U/2 324U/2 314U/2 313U/2 311N/2 290U/2 289U/2 282N
                                            266U   264P   256U/2 251U   249P   241U/2 235N   232U/2 230N
                                            218U/2 216N   167N   110U   106U    31P
   10 MINV      I  VARIABLE     POINTEE     365U   326U/2 324U/2 310N/2 290U/2 289U/2 281N   266U   264P
                                            256U/2 251U   249P   241U/2 234N   229N   215N   165N   111U
                                            108U    32P
    7 MPLOT     I  1DIM ARRAY   GRFCOM      131U    56D    55S    47D
    0 MSLICE    I  VARIABLE     GRFCOM      424P   420P   416U   415U   414U   166P    60P    56D    55S
```

```
    0 NBU      I  VARIABLE     NBUR     350N   338N   46D
      NBURST   I  PARAMETER              50P/2  44P/2  43P/2  41S
      NDM1     I  PARAMETER             419N   326P/2 324P/2 307P/2 305P/2 290P   289P   266P   260P
                                        256P   251P   245P   241P   120U   38P/3  37P/3  34P/3  33P/3
                                         28P/2  26P/2  24P/2   8P/3   7P    6P/3   5S
      NDM12    I  PARAMETER              23P/2  21S
      NDM1P2   I  PARAMETER             407P/2 385N   377N   376U   27P/2  22S
      NDM2     I  PARAMETER             418N   124U   123U   34P/2  33P/3  28P/2  26P/2  24P/2   8P/3
                                          7P    6P/3   5S
 1576 NH       I  VARIABLE              424U   423U   422U   421U   420U   419I   312U/5 311I   233U/5
                                        292I   285U/3 284U/3 283U/2 282I   236U/2 235I   231U/9 230I
                                        217U/9 216I   169U/4 168U/2 167I
 1553 NI       I  VARIABLE              402U   401U   374U   372N   369U   368N   364U   359N   114U
                                        112S
 1555 NIJ      I  VARIABLE              403U   381U   380U   379U   378U   377N   376U   372N   370U
                                        114S
 1554 NJ       I  VARIABLE              363N   114U   113S
    4 NLABEL   I  VARIABLE     DUM.ARG. 154P   141P   58U    58P    1D
      NNNX     I  PARAMETER              12P    2S
      NNNX2    I  PARAMETER               4S
      NNNXP    I  PARAMETER              17P   16P    15P    3S
      NNNY     I  PARAMETER              15P    2S
      NNNY2    I  PARAMETER              17P   16P     4S
      NNNYP    I  PARAMETER               3S
      NNNZ     I  PARAMETER              15P    2S
      NNNZ2    I  PARAMETER              17P   16P    13P/4  11P/3   4S
      NNNZP    I  PARAMETER               3S
      NNYZ2D   I  PARAMETER              20S
      NOBRDR      EXTERNAL              138U
      NOP      I  PARAMETER              44P   43P/2  41S
  430 NOPP     I  1DIM ARRAY   PARTL    351N   344P   340N   45D    44D
      NPARTL   I  PARAMETER              42S
      NSPEC    I  PARAMETER             272U   232P   218P   10S
      NSUM     I  PARAMETER              12P   10S
 1575 NV       I  VARIABLE              424U/2 423U   422U   421U   420U/2 418I   314U/2 313U/2 312U/5
                                        310I   295U/2 294U/2 293U/5 291I   285U/3 284U/3 283U/2 281I
                                        236U/2 234I   232U/5 231U/8 229I   218U/5 217U/8 215I   169U/4
                                        168U/3 166U/2 165I
      NVAR     I  PARAMETER              23P   20S
      PAGE        EXTERNAL              137U
      PERIML      EXTERNAL              158U
      POINTS      EXTERNAL              352U
    0 PRE      R  2D EQ. ARRAY LOCOM1   314P/2 313P/2 312S   295P/2 294P/2 293S   285U   284U   283S
                                        236U   232P   218P   207P   35D    33D
47124 PSCR     R  1D EQ. ARRAY LOCOM1   232P   218P   39D    37D
      PWRIT       EXTERNAL              348U   279U   227U   213U   204U   195U   184U   175U   156U
                                        155U   154U
    0 RCAMB    R  1DIM ARRAY   HYDFIX   420P/2 166P/2 14D    11D
    2 RFSTF    CH VARIABLE     DUM.ARG.  59U   49D    1D
75700 RHA      R  2DIM ARRAY   PLTVAR   232P   218P   178P    9D     8D
112200 RHB     R  2DIM ARRAY   PLTVAR   232P   218P   189P    9D     8D
61400 RHO      R  2DIM ARRAY   PLTVAR   420S   283U   236U   232P   231U   218P   217U   169U   168U
                                          9D    8D
47014 RSCR     R  1D EQ. ARRAY LOCOM1   232P   218P   168U   166S   39D    37D
    0 RVX      R  2DIM ARRAY   PLTVAR   421S   236U/2 231U/2 217U/2 98P    68P     9D     6D
14300 RVY      R  2DIM ARRAY   PLTVAR   422S   236U/2 231U/2 217U/2 83P    69P     9D     6D
```

```
  30600 RVZ       R  2DIM ARRAY     PLIVAR      423S    236U/2  233U/2  217U/2  99P     84P     9D      6D
  47014 SCR1      R  2D EQ. ARRAY   LOCOM1      307P    305P    293U/2  289P    260P    258P    256P    245P    243P
                                                241P    40U/2   39U/3   36D     34D
  64020 SCR2      R  2D EQ. ARRAY   LOCOM1      307P    305P    293U/2  290P    40D     36D     34D
  47344 SCRA      R  1D EQ. ARRAY   LOCOM1      232P    218P    40D     38D
  47454 SCRB      R  1D EQ. ARRAY   LOCOM1      232P    218P    40D     38D
  64020 SCRC      R  1D EQ. ARRAY   LOCOM1      232P    218P    40D     38D
        SFT          EXTERNAL                   323U    304U    157U
        SHDCHR       EXTERNAL                   135U
      0 SPACE     R  2DIM ARRAY     HOLDER      390P/2  389S    388S    387S    386S    251P    245P    24D
  14302 SPACE2    R  2DIM ARRAY     HOLDER      169P/2  169S    168S    164P    24D
        SQRT      R  EXTERNAL                   317U    316U    298U    297U    236U
        SWISSM       EXTERNAL                   134U
        THKFRM       EXTERNAL                   146U
      3 TIME      R  VARIABLE       GLOVAR      59U     19D
   1577 TLENG     R  VARIABLE                   337P    336S    277P    276S    225P    224S    211P    210S    202P
                                                201S    193P    192S    182P    181S    173P    172S
    776 V         R  2DIM ARRAY                 407P    403P    402P    389U    388U    379U    379S    378U    378S
                                                370U    370S    369U    369S    366U    366S    365S    27D
      3 VCOR      R  1DIM ARRAY     POINTEE     365U    290P    289P    256P    241P    109U    108U    29P     26D
   1601 VELMAX    R  VARIABLE                   326P    324P    318U    316P    316S    313P    313S    307P    305P
                                                299U    297P    297S    294P    294S    286S
   1602 VELMIN    R  VARIABLE                   326P    324P    318U/2  318S    317P    317S    314P    314S    307P
                                                305P    299U/2  299S    298P    298S    295P    295S    287S
        VELVCT       EXTERNAL                   326U    324U    307U    305U
   1566 VHI       R  VARIABLE                   399P    157P    128P
   1565 VLO       R  VARIABLE                   398P    157P    128P
      3 VMAX      R  VARIABLE       BOUND       336U    323P    321U    304P    302U    276U    224U    210U    201U
                                                192U    181U    172U    157P    150P    148U    139U    128P    119U
                                                109S    57D
      2 VMIN      R  VARIABLE       BOUND       336U    323P    321U    304P    302U    276U    224U    210U    201U
                                                192U    181U    172U    157P    150P    148U    139U    128P    113U
                                                108S    57D
        WSIZE        EXTERNAL                   128U
   1603 XA        R  VARIABLE                   323P    322U    321S    304P    303U    302S
   1604 XB        R  VARIABLE                   323P    322S    304P    303S
      0 XCOR      R  1DIM ARRAY     GRDVAR      96P     88U     86U     66P     18D     15D
        XMESS     R  EXTERNAL                   336U    276U    224U    210U    201U    192U    181U    172U
        XNAME        EXTERNAL                   144U
      0 XP        R  2DIM ARRAY     PARTL       353U    352P    344P    341U    341S    45D     43D
   1572 XSTEP     R  VARIABLE                   150P    149S
    111 YCOR      R  1DIM ARRAY     GRDVAR      103U    101U    81P     67P     18D     15D
        YNAME        EXTERNAL                   145U
   1571 YSTEP     R  VARIABLE                   150P    149U    148S
    152 ZCOR      R  1DIM ARRAY     GRDVAR      97P     82P     73U     71U     18D     15D
    214 ZP        R  2DIM ARRAY     PARTL       353U    352P    344P    342U    342S    45D     43D
TABLE OF EXTERNAL NAMES
-----------------------

        $EFA         EXTERNAL                   59U     58U
        $EFF         EXTERNAL                   334U    274U    222U    208U    199U    190U    179U    170U    103U
                                                101U    88U     86U     73U     71U     59U     58U
        $EFI         EXTERNAL                   334U    274U    222U    208U    199U    190U    179U    170U    103U
                                                101U    88U     86U     73U     71U     59U     59U
        $EFV         EXTERNAL                   190U    179U    103U/2  101U/2  88U/2   86U/2   73U/2   71U/2   59U/2
```

$RNL		EXTERNAL	412U								
$WFI		EXTERNAL	353U	349U							
$WFL		EXTERNAL	353U	349U							
$WFV		EXTERNAL	353U/3								
$WNL		EXTERNAL	413U								
AGSETF		EXTERNAL	406U	405U	404U	403U	402U	401U	400U	399U	398U
			397U	396U							
AGSETI		EXTERNAL	395U								
ANOTAT		EXTERNAL	394U								
AREA2D		EXTERNAL	139U								
COLCON		EXTERNAL	264U	258U	249U	243U					
CONINF		EXTERNAL	290U	289U	256U	241U					
CONREC		EXTERNAL	266U	260U							
CURVE		EXTERNAL	390U	344U							
DISCON		EXTERNAL	251U	245U							
ENDPL		EXTERNAL	392U	346U	330U						
EOS		EXTERNAL	232U	218U							
EZMXY		EXTERNAL	407U								
FRAME		EXTERNAL	147U								
FRAMEN		EXTERNAL	355U	329U	269U						
GDONE		EXTERNAL	433U								
GPLOT		EXTERNAL	425U								
GRAF		EXTERNAL	150U								
HEADIN		EXTERNAL	143U	142U	141U						
HEIGHT		EXTERNAL	140U								
ISMAX	I	EXTERNAL	313U	294U							
ISMIN	I	EXTERNAL	314U	295U							
LIBDISP		EXTERNAL	427U								
LIBNCAR		EXTERNAL	429U	320U	301U						
MARKER		EXTERNAL	339U								
MESSAG		EXTERNAL	337U	277U	225U	211U	202U	193U	182U	173U	
NOBRDR		EXTERNAL	138U								
PAGE		EXTERNAL	137U								
PERIML		EXTERNAL	158U								
POINTS		EXTERNAL	352U								
PWRIT		EXTERNAL	348U	279U	227U	213U	204U	195U	184U	175U	156U
			155U	154U							
SET		EXTERNAL	323U	304U	157U						
SHDCHR		EXTERNAL	135U								
SQRT	R	EXTERNAL	317U	316U	298U	297U	236U				
SWISSM		EXTERNAL	134U								
THKFRM		EXTERNAL	146U								
VELVCT		EXTERNAL	326U	324U	307U	305U					
WSIZE		EXTERNAL	128U								
XMESS	R	EXTERNAL	336U	276U	224U	210U	201U	192U	181U	172U	
XNAME		EXTERNAL	144U								
YNAME		EXTERNAL	145U								

ABBREVIATIONS USED ABOVE (THESE ARE KEYED TO THE SOURCE LISTING LINE NUMBER)

```
A  USED IN FORTRAN  ASSIGN STATEMENT      P  USED IN CALL/FUNC CALL OR ARRAY DEF
D  DEFINED IN  DECLARATIVE STATEMENT      R  FORMAT  USED  IN A  READ  STATEMENT
E  STATEMENT NUMBER ENDING A DO LOOP      S  STORED  SO CONTENTS  MAY BE CHANGED
I  INDEX OF A DO  OR IMPLIED DO LOOP      U  NAME  USED IN EXECUTABLE  STATEMENT
J  STATEMENT NUMBER USED IN TRANSFER      W  FORMAT  USED IN A  WRITE  STATEMENT
L  SOURCE LINE OF A STATEMENT NUMBER      *  DEFINED  OR DECLARED  BUT NOT  USED
N  NAME USED AS A  DO LOOP PARAMETER      ?  TEN OR  MORE REFERENCES  TO  SYMBOL
```

TABLE OF PARAMETERS ENCOUNTERED

NRURST	= 2	NDM1	= 72
NDM12	= 6660	NDM1P2	= 160
NDM2	= 88	NNNX	= 72
NNNX2	= 74	NNNXP	= 73
NNNY	= 33	NNNY2	= 35
NNNYP	= 34	NNNZ	= 88
NNNZ2	= 90	NNNZP	= 89
NNYZ2D	= 3150	NOP	= 70
NPARTL	= 422	NSPEC	= 1
NSUM	= 5	NVAR	= 5

TABLE OF LOOPS ENCOUNTERED

LABEL	INDEX	FROM	TO	ADDRESS	LENGTH
230	IP	130	409	2175d	2341
72	NV	165	169	2374a	36
72	NH	167	169	2415b	12
130	NV	215	219	2677b	73
120	NH	216	217	2707d	22
146	NV	229	233	3035c	73
145	NH	230	231	3046a	21
150	NV	234	236	3134a	40
150	NH	235	236	3144c	25
180	NV	281	285	3423a	32
180	NH	282	285	3436c	14
200	NV	291	296	3526b	44
190	NH	292	293	3535c	11
220	NV	310	315	3661b	56
210	NH	311	312	3675c	10
405	K	338	345	4076b	40
406	IJK	340	343	4113b	11
410	K	350	354	4160a	37
410	I	351	354	4165b	26
421	I	359	362	4232a	7
422	J	363	367	4253d	11
423	I	368	371	4302d	10
424	J	372	375	4332d	6
425	J	377	382	4355c	14

```
    426 J            385   391    4411c      16
    240 NV           418   424    4575b      34
    240 NH           419   424    4611b      15
 BLOCK NAMES AND LENGTHS IN OCTAL

    4655 CONV3D         65 #1B          1243 #CL        126500 PLTVAR      1741 HYDFIX      1116 GRDVAR
      14 GLOVAR      30602 HOLDER     116030 LOCOM1        646 PARTI          1 NBUR          35 GRFCOM
       4 BOUND
STATIC SPACE (IN OCTAL)
--- --------------- ---
B SAVE:                 47
I SAVE:                 10
CONSTANTS:             274
VARIABLES:            1316
TEMPORARIES:          1251
CODE:                 3043
                   -------
TOTAL:                6205
```

```
      subroutine disprf (arrx,arry,length)
      real arrx(length), arry(length)
c
c  ***************************************************************
c  *   This Subroutine plots profile graphs from the input data.    *
c  *    It is called to do profiles on part of an array, using       *
c  *   DISSPLA calls to the routine curve.                          *
c  ***************************************************************
c
      xmax=arrx(1)
      xmin=arrx(1)
      ymax=arry(1)
      ymin=arry(1)
c   calculate the maximum and minimum x and y values
      do 10 icount=1,length
      if (arrx(icount) .lt. xmin) xmin=arrx(icount)
      if (arry(icount) .lt. ymin) ymin=arry(icount)
      if (arrx(icount) .gt. xmax) xmax=arrx(icount)
      if (arry(icount) .gt. ymax) ymax=arry(icount)
 10   continue
      if (ymax .gt. 0.0) then
        ymax=ymax*1.02
      else
        ymax=ymax*0.98
      endif
      if (ymin .gt. 0.0) then
        ymin=ymin*0.98
      else
        ymin=ymin*1.02
      endif
      if ((ymax .eq. ymin) .and. (ymin .eq. 0.)) then
        ymax=1.0
        ymin=-1.0
      endif
      call thkfrm (.02)
      call frame
c   set up the axes with an appropriate range.
c   then draw the curve.
      call graf (xmin,'scale',xmax,ymin,'scale',ymax)
      call curve (arrx,arry,length,0)
      call endpl (0)
      return
      end
```

```
 TABLE OF STATEMENT NUMBERS (ALL ADDRESSES IN TABLES ARE IN OCTAL)
  ----- -- ---------
  NUMBER USE    SOURCE PROGRAM REFERENCES

     10 UNDEF**     12I        7I
     10A36d          7I
     10B55a          7I
  00002 UNDEF**      8I
  00003 UNDEF**      9I
  00004 UNDEF**     10L
  00005 UNDEF**     11L
  00006 60a         13L
  00007 61d         15L
  00008 64d         18L
  00009 66c         20I
  00010 74a         23L
   (SN=STATEMENT NUMBER, GSN=GENERATED STATEMENT NUMBER)
   (FN=FORMAT NUMBER, UNDEF**=UNDEFINED STATEMENT NUMBER)
 TABLE OF NAMES ENCOUNTERED (ADDRESS FOR DUMMY ARGUMENT IS THE ARGUMENT NUMBER)
 ---------------------- ---

  ADDRESS NAME     TYPE MAIN USAGE   BLOCK        SOURCE PROGRAM REFERENCES

        1 ARRX      R   1DIM ARRAY   DUM.ARG.       30P     10U/2    8U/2    4U     3U     2D     1D
        2 ARRY      R   1DIM ARRAY   DUM.ARG.       30P     11U/2    9U/2    6U     5U     2P     1D
          CURVE         EXTERNAL                    30U
       16 DISPRF        ENTRY                        1D/2
          ENDPI         EXTERNAL                    31U
          FRAME         EXTERNAL                    28U
          GRAF          EXTERNAL                    29U
       14 ICOUNT    I   VARIABLE                    11U/2   10U/2    9U/2    8U/2   7I.
        3 LENGTH    I   VARIABLE     DUM.ARG.       30P      7N      2P/2    1D
          MAIN.         ENTRY
          THKERM        EXTERNAL                    27U
       10 XMAX      R   VARIABLE                    29P     10S     10U      3S
       11 XMIN      R   VARIABLE                    29P      8S      8U      4S
       12 YMAX      R   VARIABLE                    29P     24S     23U     16U    16S    14U    14S    13U    11S
                                                    11U      5S
       13 YMIN      R   VARIABLE                    29P     25S     23U/2   21U    21S    19U    19S    18U     9S
                                                     9U      6S
 TABLE OF EXTERNAL NAMES
 ----------------------

          CURVE         EXTERNAL                    30U
          ENDPI         EXTERNAL                    31U
          FRAME         EXTERNAL                    28U
          GRAF          EXTERNAL                    29U
          THKERM        EXTERNAL                    27U
```

```
ABBREVIATIONS USED ABOVE (THESE ARE KEYED TO THE SOURCE LISTING LINE NUMBER)

A  USED IN FORTRAN ASSIGN STATEMENT       P  USED IN CALL/FUNC CALL OR ARRAY DEF
D  DEFINED IN DECLARATIVE STATEMENT       R  FORMAT USED IN A READ STATEMENT
F  STATEMENT NUMBER ENDING A DO LOOP      S  STORED SO CONTENTS MAY BE CHANGED
I  INDEX OF A DO OR IMPLIED DO LOOP       U  NAME USED IN EXECUTABLE STATEMENT
J  STATEMENT NUMBER USED IN TRANSFER      W  FORMAT USED IN A WRITE STATEMENT
L  SOURCE LINE OF A STATEMENT NUMBER      *  DEFINED OR DECLARED BUT NOT USED
N  NAME USED AS A DO LOOP PARAMETER       ?  TEN OR MORE REFERENCES TO SYMBOL

TABLE OF LOOPS ENCOUNTERED

LABEL INDEX      FROM -  TO   ADDRESS - LENGTH

   10 ICOUNT       7     12       36d        17
BLOCK NAMES AND LENGTHS IN OCTAL

    120-DISPRF            23-#IB              21-#CL
STATIC SPACE (IN OCTAL)
-----------------------
B SAVE:                  16
T SAVE:                   5
CONSTANTS:                6
VARIABLES:                7
TEMPORARIES:             21
CODE:                   103
                    -------
TOTAL:                  164
```

```
947
948     1.           subroutine distpl (arrx,arry,length,space)
949     2.           real arrx(length), arry(length)
950     3.           real space(length)
951        c
952        c  ***************************************************************
953        c  *  This Subroutine plots profile graphs from the input data.    *
954        c  *   It is used for plots versus distance, changing from cm to km. *
955        c  ***************************************************************
956     4.           do 5 icount=1,length
957     5.    5      space(icount)=arrx(icount)/100000.
958        c
959     6.           xmax=space(1)
960     7.           xmin=space(1)
961     8.           ymax=arry(1)
962     9.           ymin=arry(1)
963        c     calculate maximum and minimum x and y values
964    10.           do 10 icount=1,length
965    11.           if (space(icount) .lt. xmin) xmin=space(icount)
966    12.           if (arry(icount) .lt. ymin) ymin=arry(icount)
967    13.           if (space(icount) .gt. xmax) xmax=space(icount)
968    14.           if (arry(icount) .gt. ymax) ymax=arry(icount)
969    15.   10      continue
970    16.           if (ymax .gt. 0.0) then
971    17.             ymax=ymax*1.02
972    18.           elseif (ymax .lt. 0.0) then
973    19.             ymax=ymax*0.98
974    20.           endif
975    21.           if (ymin .gt. 0.0) then
976    22.             ymin=ymin*0.98
977    23.           elseif (ymin .lt. 0.0) then
978    24.             ymin=ymin*1.02
979    25.           endif
980    26.           if ((ymax .eq. ymin) .and. (ymin .eq. 0.)) then
981    27.             ymax=1.0
982    28.             ymin=-1.0
983    29.           endif
984    30.           call thkfrm (.02)
985    31.           call frame
986        c     set up axes with an appropriate range of values,
987        c     then draw the curve.
988    32.           call graf (xmin,'scale',xmax,ymin,'scale',ymax)
989    33.           call curve (space,arry,length,0)
990    34.           call endpl (0)
991    35.           return
992    36.           end
DISTPL           VECTOR LOOP BEGINS AT SEQ. NO.          4, P=       23b
```

```
TABLE OF STATEMENT NUMBERS (ALL ADDRESSES IN TABLES ARE IN OCTAL)
---- -----   ------
  NUMBER USE    SOURCE PROGRAM REFERENCES

     5 UNDEF**       5L      4I
     5A35b           4I
     5B43d           4L
    10 UNDEF**      15L     10F
    10A57d          10I
    10B76a          10I
 00002 UNDEF**      11L
 00003 UNDEF**      12I
 00004 UNDEF**      13I
 00005 UNDEF**      14L
 00006 101a         16L
 00007 103d         18I
 00008 103d         18L
 00009 106d         21L
 00010 111c         23L
 00011 111c         23L
 00012 117a         26L
  (SN=STATEMENT NUMBER, GSN=GENERATED STATEMENT NUMBER)
  (FN=FORMAT NUMBER, UNDEF**=UNDEFINED STATEMENT NUMBER)
TABLE OF NAMES ENCOUNTERED (ADDRESS FOR DUMMY ARGUMENT IS THE ARGUMENT NUMBER)
---------------------------

 ADDRESS NAME     TYPE MAIN USAGE  BLOCK        SOURCE PROGRAM REFERENCES

       1 ARRX      R   1DIM ARRAY  DUM.ARG.       5U     2D     1D
       2 ARRY      R   1DIM ARRAY  DUM.ARG.      33P    14U/2  12U/2  9U     8U     2P     1D
         CURVE         EXTERNAL                  33U
      17 DISTPL        ENTRY                      1D/2
         ENDPL         EXTERNAL                  34U
         FRAME         EXTERNAL                  31U
         GRAF          EXTERNAL                  32U
      11 ICOUNT    I   VARIABLE                  14U/2  13U/2  12U/2  11U/2  10I    5U/2   4I
       3 LENGTH    I   VARIABLE    DUM.ARG.      33P    10N    4N     3P     2P/2   1D
         MAIN.         ENTRY
       4 SPACE     R   1DIM ARRAY  DUM.ARG.      33P    13U/2  11U/2  7U     6U     5S     3D     1D
         THKFRM        EXTERNAL                  30U
      12 XMAX      R   VARIABLE                  32P    13S    13U    6S
      13 XMIN      R   VARIABLE                  32P    11S    11U    7S
      14 YMAX      R   VARIABLE                  32P    27S    26U    19U    19S    18U    17U    17S    16U
                                                 14S    14U    8S
      15 YMIN      R   VARIABLE                  32P    28S    26U/2  24U    24S    23U    22U    22S    21U
                                                 12S    12U    9S
TABLE OF EXTERNAL NAMES
--------------- -------

         CURVE         EXTERNAL                  33U
         ENDPL         EXTERNAL                  34U
```

```
     FRAME        EXTERNAL                        31U
     GRAF         EXTERNAL                        32U
     THKFRM       EXTERNAL                        30U
    ABBREVIATIONS USED ABOVE (THESE ARE KEYED TO THE SOURCE LISTING LINE NUMBER)
    -- ---- ---- --- ----
    A  USED IN FORTRAN  ASSIGN STATEMENT     P  USED IN CALL/FUNC CALL OR ARRAY DEF
    D  DEFINED IN  DECLARATIVE STATEMENT     R  FORMAT  USED  IN A  READ  STATEMENT
    E  STATEMENT NUMBER ENDING A DO LOOP     S  STORED  SO CONTENTS  MAY BE CHANGED
    I  INDEX OF A DO  OR IMPLIED DO LOOP     U  NAME  USED IN EXECUTABLE  STATEMENT
    J  STATEMENT NUMBER USED IN TRANSFER     W  FORMAT  USED IN A  WRITE  STATEMENT
    L  SOURCE LINE OF A STATEMENT NUMBER     *  DEFINED  OR DECLARED  BUT NOT  USED
    N  NAME USED AS A  DO LOOP PARAMETER     ?  TEN OR  MORE REFERENCES  TO  SYMBOL

 TABLE OF LOOPS ENCOUNTERED
 -------------- ----------
  LABEL INDEX      FROM -  TO   ADDRESS - LENGTH

      5 ICOUNT        4     5       35b        6
     10 ICOUNT       10    15       57d       17
  BLOCK NAMES AND LENGTHS IN OCTAL

      143-DISTPL            30-#IB            21 #CL
 STATIC SPACE (IN OCTAL)
 -----------------------
 B SAVE:                   23
 T SAVE:                    5
 CONSTANTS:                 7
 VARIABLES:                 7
 TEMPORARIES:              21
 CODE:                    125
                     --------
 TOTAL:                   214
```

```
  1.          subroutine discon (conarr,mdim,cnhold,kh,kv)
  2.          real conarr (mdim,kv), cnhold (kh,kv)
  3.          common work (5000)
        c
        c ****************************************************************
        c *     This Subroutine Draws contour plots using DISSPLA calls,  *
        c *     plotting the specified part of the array                  *
        c ****************************************************************
        c
  4.          do 10 icount=1,kh
  5.          do 10 jcount=1,kv
  6.          cnhold(icount,jcount)=conarr(icount,jcount)
  7.     10   continue
        c   set up the contour levels
  8.          call bcomon (5000)
  9.          call conmak (cnhold,kh,kv,'scale')
        c   specify the appearance of the lines, and their labels
 10.          call height (.10)
 11.          call conlin (0,'solid','labels',1,3)
 12.          call conang (60.)
 13.          call conthn (0.06)
        c   draw the contour lines
 14.          call contur (1,'labels','draw')
 15.          call endpl(0)
 16.          return
 17.          end
DISCON      VECTOR LOOP BEGINS AT SEQ. NO.        5, P=      35d
```

```
TABLE OF STATEMENT NUMBERS (ALL ADDRESSES IN TABLES ARE IN OCTAL)
---------------------------

  NUMBER USE    SOURCE PROGRAM REFERENCES

    10 UNDEF**       7I        5I        4E
    10A35d           4I
    10B62c           4I
    10C51b           5I
    10D57d           5L
  (SN=STATEMENT NUMBER, GSN=GENERATED STATEMENT NUMBER)
  (FN=FORMAT NUMBER, UNDEF**=UNDEFINED STATEMENT NUMBER)
TABLE OF NAMES ENCOUNTERED (ADDRESS FOR DUMMY ARGUMENT IS THE ARGUMENT NUMBER)
--------------------------

ADDRESS NAME     TYPE MAIN USAGE  BLOCK        SOURCE PROGRAM REFERENCES

        BCOMON        EXTERNAL                   8U
      3 CNHOLD   R    2DIM ARRAY  DUM.ARG.       9P       6S       2P       1D
        CONANG        EXTERNAL                  12U
      1 CONARR   R    2DIM ARRAY  DUM.ARG.       6U       2D       1D
        CONLIN        EXTERNAL                  11U
        CONMAK        EXTERNAL                   9U
        CONTHN        EXTERNAL                  13U
        CONTUR        EXTERNAL                  14U
     24 DISCON        ENTRY                      1D/2
        ENDPL         EXTERNAL                  15U
        HEIGHT        EXTERNAL                  10U
     21 ICOUNT   I    VARIABLE                   6U/2     4I
     22 JCOUNT   I    VARIABLE                   6U/2     5I
      4 KH       I    VARIABLE    DUM.ARG.       9P       4N       2P       1D
      5 KV       I    VARIABLE    DUM.ARG.       9P       5N       2P/2     1D
        MAIN.         ENTRY
TABLE OF EXTERNAL NAMES
-----------------------

        BCOMON        EXTERNAL                   8U
        CONANG        EXTERNAL                  12U
        CONLIN        EXTERNAL                  11U
        CONMAK        EXTERNAL                   9U
        CONTHN        EXTERNAL                  13U
        CONTUR        EXTERNAL                  14U
        ENDPL         EXTERNAL                  15U
        HEIGHT        EXTERNAL                  10U
```

```
        ABBREVIATIONS USED ABOVE (THESE ARE KEYED TO THE SOURCE LISTING LINE NUMBER)

        A  USED IN FORTRAN  ASSIGN STATEMENT      P  USED IN CALL/FUNC CALL OR ARRAY DEF
        D  DEFINED IN  DECLARATIVE STATEMENT      R  FORMAT  USED  IN A  READ  STATEMENT
        E  STATEMENT NUMBER ENDING A DO LOOP      S  STORED  SO CONTENTS  MAY BE CHANGED
        I  INDEX OF A DO  OR IMPLIED DO LOOP      U  NAME  USED IN EXECUTABLE  STATEMENT
        J  STATEMENT NUMBER USED IN TRANSFER      W  FORMAT  USED IN A  WRITE  STATEMENT
        L  SOURCE LINE OF A STATEMENT NUMBER      *  DEFINED  OR DECLARED  BUT NOT  USED
        N  NAME USED AS A, DO LOOP PARAMETER      ?  TEN OR  MORE REFERENCES  TO  SYMBOL

 TABLE OF LOOPS ENCOUNTERED
 -------------------------
  LABEL INDEX      FROM -  TO   ADDRESS - LENGTH

     10 ICOUNT        4     7       35d       26
     10 JCOUNT        5     7       51b        6
  BLOCK NAMES AND LENGTHS IN OCTAL

      124-DISCON            31-#1B            31-#CL          11610-//
 STATIC SPACE (IN OCTAL)
 ------------------------
 B SAVE:                 27
 I SAVE:                  2
 CONSTANTS:              17
 VARIABLES:               4
 TEMPORARIES:            31
 CODE:                  101
                    -------
 TOTAL:                 206
```

```
   1           subroutine prof3d (istep,restf,label,nlabel,iplane)
         c-------------------------------------------------------------
         c     draws profiles for fast3d.
         c     Prof3d is the routine which draws the profile plots.  At present it
         c can produce seven distinct plots, those being (in order) density,
         c energy, dynamic pressure, pressure (absolute), renormalized density,
         c renormalized pressure, and z velocity.  These plots are specified by
         c non-zero values in the array mplot(1) to mplot (7).
         c     These profiles are drawn in a plane specified by the absolute value
         c of mplot.  If the value is negative, it will plot in the "vertical"
         c direction, if positive in the "horizontal" direction.  That is, if the
         c value of mplot is -36 it will plot quantity vs. distance in the z direction
         c at x = 36.
         c     These plots are fairly consistent in the way they are implemented.
         c Interpolation is handled as it is in conv3d, through the flag lintp and
         c the routine conint.  In DISSPLA, the curves are actually plotted in the
         c routine distpl, rather than in prof3d itself.
         c------------------------ array dimensions ---------------------------
   2.          parameter (nnnx=72, nnny=33, nnnz=88)
   3.          parameter (nnnxp=nnnx+1, nnnyp=nnny+1, nnnzp=nnnz+1)
   4.          parameter (nnnx2=nnnx+2, nnny2=nnny+2, nnnz2=nnnz+2)
         c------------------------ plot variables -----------------------------
   5.          parameter (ndm1=72, ndm2=88)
   6.          real rvx(ndm1,ndm2), rvy(ndm1,ndm2), rvz(ndm1,ndm2)
   7.          real erg(ndm1,ndm2)
   8.          real rho(ndm1,ndm2), rha(ndm1,ndm2), rhb(ndm1,ndm2)
   9.          common /pltvar/ rvx, rvy, rvz, erg, rho, rha, rhb
         c-------------------- grid arrays ------------------------------------
  10.          real xcor(nnnxp), ycor(nnny), zcor(nnnz)
  11.          real twodx(nnnxp), twody2(nnny2), twodz2(nnnz2)
  12.          real dtodx(nnnxp), dtody2(nnny2), dtodz2(nnnz2)
         c
  13.          common /holder/ space(ndm1,ndm2),valmin,valrng,
      +           space2(ndm1,ndm2)
  14.          common /grdvar/ xcor, ycor, zcor, twodx, twody2, twodz2, dtodx
      1        , dtody2, dtodz2
  15.          real rcamb(nnnz2),pcamb(nnnz2)
  16.          common /hydfix/ rcamb,pcamb
         c------------------------ global variables ---------------------------
  17.          common /glovar/ dtmin, dtmax, cour, time, dt, dth, rlos, nfile
      1        , nstep, lllx, llly, lllz
         c---------------------------- scratch space --------------------------
  18.          parameter (nnyz2d=nnny2+nnnz2, nspec=1, nvar=4+nspec)
  19.          parameter (ndm12=(ndm1+2)+(ndm2+2))
  20.          real dumv(ndm12,nvar), dumy(ndm12)
  21.          common /locom1/ dumv, dumy
         c------------------------------ pointers -----------------------------
  22.          real arr(ndm1,ndm2), hcor(ndm1), vcor(ndm2)
  23.          pointer (iparr,arr), (iphcor,hcor), (ipvcor,vcor)
  24.          pointer (ipminh,minh), (ipmaxh,maxh)
  25.          pointer (ipminv,minv), (ipmaxv,maxv)
         c------------------------ local declarations -------------------------
  26.          parameter (nhv=88)
  27.          real pre(ndm1,ndm2), mac(ndm1,ndm2), gam(ndm1,ndm2)
  28.          real dpr(ndm1,ndm2), rint(nhv), scrh(nhv)
```

```
      equivalence (pre,dumv), (mac,dumv(1,2)), (gam,dumv(1,3))
      equivalence (dpr,dumv(1,4)), (eint,dumv(1,5)), (scrh,dumy)
      real rscr(ndm1), pscr(ndm1), qscr(ndm1)
      real scra(ndm1), scrb(ndm1), scrc(ndm1)
      equivalence (rscr,mac), (pscr,mac(1,2)), (qscr,mac(1,3))
      equivalence (scra,mac(1,4)), (scrb,mac(1,5)), (scrc,mac(1,6))
c
      integer mplot(14)
      character *40 label
      character *8 restf
      real glab1(7), glab2(7), glab3(7)
      logical displa, darkhd
      common /qrfcom/ mslice, imin, imax, jmin, jmax, kmin, kmax,
     1    mplot, lintp, colour, dmin, dmax, displa, i1, j1, darkhd
c
c     first print labels.  use encode to set up the titles
      encode (nlabel,310,glab1) label(1:nlabel)
      encode (40,320,glab2) time,istep,restf
c
c     set window sizes.
      go to (10,20,30), mslice
c
c     X-Y slice.
   10 continue
      ipminh=loc(imin)
      ipmaxh=loc(imax)
      ipminv=loc(jmin)
      ipmaxv=loc(jmax)
      iphcor=loc(xcor)
      ipvcor=loc(ycor)
      go to 40
c
c     Y-Z slice.
   20 continue
      ipminh=loc(jmin)
      ipmaxh=loc(jmax)
      ipminv=loc(kmin)
      ipmaxv=loc(kmax)
      iphcor=loc(ycor)
      ipvcor=loc(zcor)
      go to 40
c
c     X-Z slice.
   30 continue
      ipminh=loc(imin)
      ipmaxh=loc(imax)
      ipminv=loc(kmin)
      ipmaxv=loc(kmax)
      iphcor=loc(xcor)
      ipvcor=loc(zcor)
   40 continue
c
c     plot profiles.
      ispec=1
      do 240 ip=1,7
      if (mplot(ip).eq.0) go to 240
```

```
c     make DISSPLA calls to set the page size, print type, plot area,
c     and draw the titles, or NCAR calls to write the titles.
 50      if (displa) then
           call page (10.,11.)
           call nobrdr
           call area2d (6.,8.)
           if (darkbd) then
             call shdchr (90.,1,.003,1)
             call swissm
           endif
           call height (.16)
           call headin (glab1,nlabel-1,1.5,3)
           call headin (glab2,25,1.5,3)
         else
           call pwrit (550,50,glab1,nlabel,2,0,0)
           call pwrit (550,25,glab2,40,1,0,0)
         endif
c
         go to (60,100,110,130,1000,1010,1020), ip
c
c     density profile.
c     plots up to 3 species of absolute density, usually total density (1),
c     dust density (2), and explosive density (3).
 60      go to (70,80,90), ispec
c     label the axes
 70      if (displa) then
           call xname ('distance (km)',13)
           call yname ('density (g/cm**3)',17)
         else
           call anotat (14hdistance, km.$,15hdensity, gm/cc$,1,1,0,0)
         endif
c     set pointer to array
         iparr=loc(rho)
         go to 160
c     label the axes
 80      if (displa) then
           call xname ('distance (km)',13)
           call yname ('density 2 (g/cm**3)',19)
         else
           call anotat (14hdistance, km.$,17hdensity 2, gm/cc$,1,1,0,0)
         endif
c     set pointer to array
         iparr=loc(rha)
         go to 160
c     label the axes
 90      if (displa) then
           call xname ('distance (km)',13)
           call yname ('density 3 (g/cm**3)',19)
         else
           call anotat (14hdistance, km.$,17hdensity 3, gm/cc$,1,1,0,0)
         endif
c     set pointer to array
         iparr=loc(rhb)
         go to 160
c
c     energy profile
```

```
           c     label the axes
     112.   100     if (displa) then
     113              call xname ('distance (km)',13)
     114.             call yname ('energy (ergs/cm**3)',19)
     115.           else
     116.             call anotat (14hdistance, km.$,16henergy, ergs/cc$,1,1,0,0)
     117.           endif
           c     set pointer to array
     118.           iparr=loc(erg)
     119.           go to 160
           c
           c     dynamic pressure profile.
           c     label the axes
     120.   110     if (displa) then
     121.             call xname ('distance (km)',13)
     122.             call yname ('dynamic pressure (dynes/cm**2)',30)
     123.           else
     124.             call anotat(14hdistance, km.$,
                +         30hdynamic pressure, dynes/cm**2$,1,1,0,0)
     125.           endif
           c     set pointer to array
     126.           iparr=loc(dpr)
           c     calculate the dynamic pressure from momenta and density
     127.           do 120 nv=minv,maxv
     128.           do 120 nh=minh,maxh
     129.   120     dpr(nh,nv)=0.5*(rvx(nh,nv)*rvx(nh,nv)+rvy(nh,nv)*rvy(nh,nv)
                1     +rvz(nh,nv)*rvz(nh,nv))/rho(nh,nv)
     130.           go to 160
           c
           c     pressure profile.
           c     this is a profile of absolute pressure.
           c
           c     label the axes
     131.   130     if (displa) then
     132.             call xname ('distance (km)',13)
     133.             call yname ('pressure (dynes/cm**2)',22)
     134.           else
     135              call anotat (14hdistance, km.$,22hpressure, dynes/cm**2$,
                +          1,1,0,0)
     136.           endif
           c     set pointer to array
     137.           iparr=loc(pre)
           c     calculate the dynamic pressure, for use in calculating pressure
     138.           do 135 nv=minv,maxv
     139.           do 135 nh=minh,maxh
     140.   135     dpr(nh,nv)=0.5*(rvx(nh,nv)*rvx(nh,nv)+rvy(nh,nv)*rvy(nh,nv)
                1     +rvz(nh,nv)*rvz(nh,nv))/rho(nh,nv)
           c     calculate the pressure, using equations of state subroutine eos
     141.           do 150 nv=minv,maxv
     142.           do 140 nh=minh,maxh
     143.   140     eint(nh)=erg(nh,nv)-dpr(nh,nv)
     144            call eos (maxh-minh+1,nspec,rho(minh,nv),eint(minh),gam(minh
                1     ,nv),pre(minh,nv),rscr(minh),pscr(minh),qscr(minh),scra(minh)
                2     ,scrb(minh),scrc(minh),rha(minh,nv),rhb(minh,nv))
     145.   150     continue
     146.           go to 160
```

```
             c
             c  Renormalized density -- with respect to ambient density
             c      label the axes
      147   1000      if (displa) then
      148.              call xname ('distance (km)',13)
      149.              call yname ('renormalized density',22)
      150.            else
      151.              call anotat (14hdistance, cm.$,
                 1        22hrenormalized density$,1,1,0,0)
      152.            endif
             c     set pointer to array
      153.            iparr=loc(space2)
             c     renormalize the density with the ambient
      154.            do 1002 jjv=minv,maxv
      155.            rscr(jjv)=cvmgt(rcamb(iplane),rcamb(jjv+1),mslice .eq. 1)
      156.            do 1002 iih=minh,maxh
      157.            space2(iih,jjv)=rho(iih,jjv)/rscr(jjv)
      158.  1002      continue
      159.            go to 160
             c
             c  Renormalized pressure
             c      label the axes
      160.  1010      if (displa) then
      161.              call xname ('distance (km)',13)
      162.              call yname ('renormalized pressure',21)
      163.            else
      164.              call anotat (14hdistance, cm.$,
                 1        22hrenormalized pressure$,1,1,0,0)
      165.            endif
             c     set pointer to the array
      166.            iparr=loc(space2)
             c     calculate dynamic pressure to use in calculating pressure
      167.            do 1011 nv=minv,maxv
      168.            do 1011 nh=minh,maxh
      169.  1011      dpr(nh,nv)=0.5*(rvx(nh,nv)*rvx(nh,nv)+rvy(nh,nv)*rvy(nh,nv)
                 1      +rvz(nh,nv)*rvz(nh,nv))/rho(nh,nv)
             c     calculate pressure (absolute) from equations of state routine eos
      170.            do 1013 nv=minv,maxv
      171.            do 1012 nh=minh,maxh
      172.  1012      eint(nh)=erg(nh,nv)-dpr(nh,nv)
      173.            call eos (maxh-minh+1,nspec,rho(minh,nv),eint(minh),gam(minh
                 1      ,nv),pre(minh,nv),rscr(minh),pscr(minh),gscr(minh),scra(minh)
                 2      ,scrb(minh),scrc(minh),rha(minh,nv),rhb(minh,nv))
      174.  1013      continue
             c     renormalize density with the ambient
      175.            do 1015 jjv=minv,maxv
      176.            pscr(jjv)=cvmgt(pcamb(iplane),pcamb(jjv+1),mslice .eq. 1)
      177.            do 1015 iih=minh,maxh
      178.            space2(iih,jjv)=pre(iih,jjv)/pscr(jjv)
      179.  1015      continue
      180.            go to 160
             c
             c  Z velocity profile
             c      label the axes
      181   1020      if (displa) then
      182.              call xname ('distance (km)',13)
```

```
 183.           call vname ('z velocity (cm/sec)',19)
 184.         else
 185.           call anotal (14hdistance, cm.$,
          +       19hz velocity, cm/sec$,1,1,0,0)
 186.         endif
        c     set pointer to array
 187.         iparr=loc(space2)
        c     calculate z velocity from z momentum and density
 188.         do 1022 jjv=minv,maxv
 189.         do 1022 iih=minh,maxh
 190.         space2(iih,jjv)=rvz(iih,jjv)/rho(iih,jjv)
 191.  1022   continue
 192.         go to 160
        c
 193.   160   mpabs=iabs(mplot(ip))
 194.         go to (170,190,210), mslice
        c
        c     Draw the profile on the appropriate plane, drawing from the value
        c     of mslice (xy,yz,or xz plane) and mplot (direction and location
        c     of profile).  First draw the last title (specifying which plane,
        c     direction, and location), then plot the profile itself.  In DISSPLA,
        c     the actual drawing is done in the routine distpl, which converts from
        c     centimeters to kilometers and plots the proper portion of the grid.
        c     X-Y slice.
 195.   170    if (displa) then
 196.            if (mplot(ip).gt.0) then
 197.              encode(31,250,glab3)mpabs,vcor(mpabs)/100000.,
          +          lplane,zcor(lplane)/100000.
 198.              call headin (glab3,30,1.5,3)
 199.              call distpl (hcor(minh),arr(minh,mpabs),maxh-minh+1,space)
 200.            else
 201.              encode(31,260,glab3)mpabs,hcor(mpabs)/100000.,
          +          lplane,zcor(lplane)/100000.
 202.              call headin (glab3,30,1.5,3)
 203              do 180 nv=minv,maxv
 204.   180        scrh(nv)=arr(mpabs,nv)
 205.              call distpl (vcor(minv),scrh(minv),maxv-minv+1,space)
 206.            endif
 207.          else
 208.            if (mplot(ip).gt.0) then
 209.              encode(39,255,glab3)mpabs,vcor(mpabs),lplane,zcor(lplane)
 210.              call pwrit (550,1010,glab3,39,1,0,0)
 211.              call ezxy (hcor(minh),arr(minh,mpabs),maxh-minh+1,th$)
 212.            else
 213.              encode(39,265,glab3)mpabs,hcor(mpabs),lplane,zcor(lplane)
 214.              call pwrit (550,1010,glab3,39,1,0,0)
 215.              do 185 nv=minv,maxv
 216.   185        scrh(nv)=arr(mpabs,nv)
 217.              call ezxy (vcor(minv),scrh(minv),maxv-minv+1,th$)
 218.            endif
 219.          endif
 220.          go to 230
        c
        c      Y-Z slice.
 221.   190    if (displa) then
 222.            if (mplot(ip).gt.0) then
```

```
              encode(31,270,glab3)mpabs,vcor(mpabs)/100000.,
     +          lplane,zcor(lplane)/100000.
              call headin (glab3,30,1.5,3)
              call distpl (hcor(minh),arr(minh,mpabs),maxh-minh+1,space)
            else
              encode(31,280,glab3)mpabs,hcor(mpabs)/100000.,
     +          lplane,zcor(lplane)/100000.
              call headin (glab3,30,1.5,3)
              do 200 nv=minv,maxv
  200         scrh(nv)=arr(mpabs,nv)
              call distpl (vcor(minv),scrh(minv),maxv-minv+1,space)
            endif
          else
            if (mplot(ip).gt.0) then
              encode(39,275,glab3)mpabs,vcor(mpabs),lplane,zcor(lplane)
              call pwrit (550,1010,glab3,39,1,0,0)
              call ezxy (hcor(minh),arr(minh,mpabs),maxh-minh+1,1h$)
            else
              encode(39,285,glab3)mpabs,hcor(mpabs),lplane,zcor(lplane)
              call pwrit (550,1010,glab3,39,1,0,0)
              do 205 nv=minv,maxv
  205         scrh(nv)=arr(mpabs,nv)
              call ezxy (vcor(minv),scrh(minv),maxv-minv+1,1h$)
            endif
          endif
          go to 230
c
c      X-Z slice.
  210     if (displa) then
            if (mplot(ip).gt.0) then
              encode(31,290,glab3)mpabs,vcor(mpabs)/100000.,
     +          lplane,ycor(lplane)/100000.
              call headin (glab3,30,1.5,3)
              call distpl (hcor(minh),arr(minh,mpabs),maxh-minh+1,space)
            else
              encode(31,300,glab3)mpabs,hcor(mpabs)/100000.,
     +          lplane,ycor(lplane)/100000.
              call headin (glab3,30,1.5,3)
              do 220 nv=minv,maxv
  220         scrh(nv)=arr(mpabs,nv)
              call distpl (vcor(minv),scrh(minv),maxv-minv+1,space)
            endif
          else
            if (mplot(ip).gt.0) then
              encode(39,295,glab3)mpabs,vcor(mpabs),lplane,ycor(lplane)
              call pwrit (550,1010,glab3,39,1,0,0)
              call ezxy (hcor(minh),arr(minh,mpabs),maxh-minh+1,1h$)
            else
              encode(39,305,glab3)mpabs,hcor(mpabs),lplane,ycor(lplane)
              call pwrit (550,1010,glab3,39,1,0,0)
              do 225 nv=minv,maxv
  225         scrh(nv)=arr(mpabs,nv)
              call ezxy (vcor(minv),scrh(minv),maxv-minv+1,1h$)
            endif
          endif
c
```

```
  1412           c      go back and plot the other species of density, if any
  1413   272.    230       ispec=ispec+1
  1414   273.              if (ispec.le.nspec .and. ip .eq. 1) go to 50
  1415   274.    240       continue
  1416   275.              return
  1417           c
  1418           c      format statements for the titles of these plots
  1419   276.    250 format ('jy,y,kz,z=',i3,',',f5.1,',',1x,i3,',',f5.1,'$')
  1420   277.    260 format ('ix,x,kz,z=',i3,',',f5.1,',',1x,i3,',',f5.1,'$')
  1421   278.    270 format ('kz,z,ix,x=',i3,',',f5.1,',',1x,i3,',',f5.1,'$')
  1422   279.    280 format ('jy,y,ix,x=',i3,',',f5.1,',',1x,i3,',',f5.1,'$')
  1423   280.    290 format ('kz,z,jy,y=',i3,',',f5.1,',',1x,i3,',',f5.1,'$')
  1424   281.    300 format ('ix,x,jy,y=',i3,',',f5.1,',',1x,i3,',',f5.1,'$')
  1425   282.    310 format (a)
  1426   283.    255 format ('jy,y,kz,z=',i3,',',e9.3,',',1x,i3,',',e9.3,'$')
  1427   284.    265 format ('ix,x,kz,z=',i3,',',e9.3,',',1x,i3,',',e9.3,'$')
  1428   285.    275 format ('kz,z,ix,x=',i3,',',e9.3,',',1x,i3,',',e9.3,'$')
  1429   286.    285 format ('jy,y,ix,x=',i3,',',e9.3,',',1x,i3,',',e9.3,'$')
  1430   287.    295 format ('kz,z,jy,y=',i3,',',e9.3,',',1x,i3,',',e9.3,'$')
  1431   288.    305 format ('ix,x,jy,y=',i3,',',e9.3,',',1x,i3,',',e9.3,'$')
  1432   289.    320 format ('time=',f5.1,' sec, step',i5,', dump ',a)
  1433   290.        end
PROF3D                VECTOR LOOP BEGINS AT SEQ. NO.        128, P=    1031d
PROF3D                VECTOR LOOP BEGINS AT SEQ. NO.        139, P=    1113d
PROF3D                VECTOR LOOP BEGINS AT SEQ. NO.        142, P=    1152c
PROF3D                VECTOR LOOP BEGINS AT SEQ. NO.        156, P=    1265b
PROF3D                VECTOR LOOP BEGINS AT SEQ. NO.        168, P=    1345d
PROF3D                VECTOR LOOP BEGINS AT SEQ. NO.        171, P=    1404c
PROF3D                VECTOR LOOP BEGINS AT SEQ. NO.        177, P=    1474a
PROF3D                VECTOR LOOP BEGINS AT SEQ. NO.        189, P=    1554c
PROF3D                VECTOR LOOP BEGINS AT SEQ. NO.        203, P=    1612c
PROF3D                VECTOR LOOP BEGINS AT SEQ. NO.        215, P=    1742b
PROF3D                VECTOR LOOP BEGINS AT SEQ. NO.        229, P=    2071c
PROF3D                VECTOR LOOP BEGINS AT SEQ. NO.        241, P=    2221b
PROF3D                VECTOR LOOP BEGINS AT SEQ. NO.        255, P=    2350c
PROF3D                VECTOR LOOP BEGINS AT SEQ. NO.        267, P=    2500b
```

```
TABLE OF STATEMENT NUMBERS (ALL ADDRESSES IN TABLES ARE IN OCTAL)

 NUMBER USE    SOURCE PROGRAM REFERENCES

   10 530c      44L    43J
   20 543c      52L    43J
   30 556c      60L    43J
   40 571a      67L    59J    51J
   50 574c     273J    71L
   60 660a      87L    86J
   70 666a      88L    87J
   80 711b      96L    87J
   90 734c     104L    87J
  100 757d     112L    86J
  110 1003a    120L    86J
  120 UNDEF**  129L   128F   127F
  120A1031d    127L
  120B1064c    127L
  120C1042b    128L
  120D1061d    128L
  130 1065a    131L    86J
  135 UNDEF**  140L   139F   138E
  135A1113d    138L
  135B1146c    138L
  135C1124b    139L
  135D1143d    139L
  140 UNDEF**  143L   142E
  140A1164b    142L
  140B1175b    142L
  150 UNDEF**  145L   141E
  150A1152c    141L
  150B1236a    141L
  160 1601d    193L   192J   180J   159J   146J   130J   119J   111J   103J
                95J
  170 1612c    195L   194J
  180 UNDEF**  204L   203F
  180A1721b    203L
  180B1727b    203L
  185 UNDEF**  216L   215F
  185A2050c    215L
  185B2056c    215L
  190 2071c    221L   194J
  200 UNDEF**  230L   229F
  200A2200b    229L
  200B2206b    229L
  205 UNDEF**  242L   241F
  205A2327c    241L
  205B2335c    241L
  210 2350c    247L   194J
  220 UNDEF**  256L   255E
  220A2457b    255L
  220B2465b    255L
  225 UNDEF**  268L   267F
  225A2606c    267L
```

225B2614c	267L		
230 2627a	272L	246J	220J
240 2633b	274L	70J	69L
240A573a	69L		
240BUNDEF**	69L		
250 FN	276L	197R	
255 FN	283L	209R	
260 FN	277L	201R	
265 FN	284L	213R	
270 FN	278L	223R	
275 FN	285L	235R	
280 FN	279L	227R	
285 FN	286L	239R	
290 FN	280L	249R	
295 FN	287L	261R	
300 FN	281L	253R	
305 FN	288L	265R	
310 FN	282L	41R	
320 FN	289L	42R	
1000 1236c	147L	86J	
1002 UNDEF**	158L	156L	154E
1002A1265b	154L		
1002B1316c	154L		
1002C1304b	156L		
1002D1313d	156L		
1010 1317a	160L	86J	
1011 UNDEF**	169L	168F	167F
1011A1345d	167L		
1011B1400c	167L		
1011C1356b	168L		
1011D1375d	168L		
1012 UNDEF**	172L	171F	
1012A1416b	171L		
1012B1427b	171L		
1013 UNDEF**	174L	170F	
1013A1404c	170L		
1013B1470a	170L		
1015 UNDEF**	179L	177L	175F
1015A1474a	175L		
1015B1525b	175L		
1015C1513a	177L		
1015D1522c	177L		
1020 1525d	181L	86J	
1022 UNDEF**	191L	189F	188E
1022A1554c	188L		
1022B1601b	188L		
1022C1565a	189L		
1022D1576c	189L		
00002 527a	43W		
00003 631b	71L		
00004 613a	75L		
00005 650a	82L		
00006 654c	86W		
00007 664c	87W		
00008 701a	88L		
00009 707a	91L		

```
 00010 724b    96L
 00011 732b    99L
 00012 747c   104L
 00013 755c   107L
 00014 772d   112L
 00015 1000d  115L
 00016 1016a  120L
 00017 1024a  123L
 00018 1100a  131L
 00019 1106a  134L
 00020 1251c  147L
 00021 1257c  150L
 00022 1332a  160L
 00023 1340a  163L
 00024 1540d  181L
 00025 1546d  184L
 00026 1611a  194W
 00027 1742b  195L
 00028 1660c  196L
 00029 1741d  200L
 00030 2071a  207L
 00031 2007c  208L
 00032 2071a  212L
 00033 2221b  221L
 00034 2137c  222L
 00035 2220d  226L
 00036 2350a  233L
 00037 2266c  234L
 00038 2350a  238L
 00039 2500b  247L
 00040 2416c  248L
 00041 2477d  252L
 00042 2627a  259L
 00043 2545c  260L
 00044 2627a  264L
  (SN-STATEMENT NUMBER, GSN-GENERATED STATEMENT NUMBER)
  (FN=FORMAT NUMBER, UNDEF*-UNDEFINED STATEMENT NUMBER)
TABLE OF NAMES ENCOUNTERED (ADDRESS FOR DUMMY ARGUMENT IS THE ARGUMENT NUMBER)
 -------------------------

ADDRESS NAME     TYPE MAIN USAGE   BLOCK      SOURCE PROGRAM REFERENCES

        $FFA          EXTERNAL                 42U    41U
        $FFF          EXTERNAL                265U   261U   253U   249U   239U   235U   227U   223U   213U
                                              209U   201U   197U    42U    41U
        $FFI          EXTERNAL                265U   261U   253U   249U   239U   235U   227U   223U   213U
                                              209U   201U   197U    42U    41U
        $FFV          EXTERNAL                265U/4 261U/4 253U/4 249U/4 239U/4 235U/4 227U/4 223U/4 213U/4
                                              209U/4 201U/4 197U/4  42U/2
        ANDIAT        EXTERNAL                185U   164U   151U   135U   124U   116U   108U   100U    92U
        AREA2D        EXTERNAL                 74U
      1 ARR      R    2DIM ARRAY   POINTEE    268U   263P   256U   251P   242U   237P   230U   225P   216U
                                              211P   204U   199P    23P    22D
        CVMGT         INLINE FUNCT.           176U   155U
```

```
  34 DARKHD    I   VARIABLE       GRFCOM      75U    40D    39D
  31 DISPLA    I   VARIABLE       GRFCOM     247U   221U   195U   181U   160U   147U   131U   120U   112U
                                             104U    96U    88U    71U    40D    39D
     DISTPL        EXTERNAL                  257U   251U   231U   225U   205U   199U
47014 DPR      R   2D EQ. ARRAY LOCOM1       172U   169S   143U   140S   129S   126P    30D    28D
64020 FINT     R   1D EQ. ARRAY LOCOM1       173P   172S   144P   143S    30D    28D
     FOS           EXTERNAL                  173U   144U
45100 ERG      R   2DIM ARRAY     PLIVAR     172U   143U   118P     9D     7D
     EZXY          EXTERNAL                  269U   263U   243U   237U   217U   211U
32010 GAM      R   2D EQ. ARRAY LOCOM1       173P   144P    29D    27D
  413 GLAB1    R   1DIM ARRAY                 83P    80P    41S    38D
  422 GLAB2    R   1DIM ARRAY                 84P    81P    42S    38D
  431 GLAB3    R   1DIM ARRAY                266P   265S   262P   261S   254P   253S   250P   249S   240P
                                             239S   236P   235S   228P   227S   224P   223S   214P   213S
                                             210P   209S   202P   201S   198P   197S    38D
15224 GSCR     R   1D EQ. ARRAY LOCOM1       173P   144P    33D    31D
    2 HCOR     R   1DIM ARRAY     POINTEE    265U   263P   253U   251P   239U   237P   227U   225P   213U
                                             211P   201U   199P    23P    22D
     HEADIN        EXTERNAL                  254U   250U   228U   224U   202U   198U    81U    80U
     HEIGHT        EXTERNAL                   79U
     IABS      I   INLINE FUNCT.             193U
  445 IHH      I   VARIABLE                 190U/3  189I  178U/2  177I  157U/2  156I
    2 IMAX     I   VARIABLE       GRFCOM      62P    46P    40D
    1 IMIN     I   VARIABLE       GRFCOM      61P    45P    40D
  441 IP       I   VARIABLE                  273U   260U   248U   234U   222U   208U   196U   193U    86P
                                              70U    69I
  404 IPARR   *I   VARIABLE                  187S   166S   153S   137S   126S   118S   110S   102S    94S
                                              23P
  405 IPHCOR  *I   VARIABLE                   65S    57S    49S    23P
  410 IPMAXH  *I   VARIABLE                   62S    54S    46S    24P
  412 IPMAXV  *I   VARIABLE                   64S    56S    48S    25P
  407 IPMINH  *I   VARIABLE                   61S    53S    45S    24P
  411 IPMINV  *I   VARIABLE                   63S    55S    47S    25P
  406 IPVCOR  *I   VARIABLE                   66S    58S    50S    23P
  440 ISPEC    I   VARIABLE                  273U   272U   272S    87P    68S
    1 ISTEP    I   VARIABLE       DUM.ARG.    42U    1D
  444 JJV      I   VARIABLE                 190U/3  188I  178U/3 176U/2  175I  157U/3 155U/2  154I
    4 JMAX     I   VARIABLE       GRFCOM      54P    48P    40D
    3 JMIN     I   VARIABLE       GRFCOM      53P    47P    40D
    6 KMAX     I   VARIABLE       GRFCOM      64P    56P    40D
    5 KMIN     I   VARIABLE       GRFCOM      63P    55P    40D
    3 LABEL    CH  VARIABLE       DUM.ARG.    41U    36D    1D
     LOC       I   INLINE FUNCT.             187U   166U   153U   137U   126U   118U   110U   102U    94U
                                              66U    65U    64U    63U    62U    61U    58U    57U    56U
                                              55U    54U    53U    50U    49U    48U    47U    46U    45U
    5 LPLANE   I   VARIABLE       DUM.ARG.  265U/2 261U/2 253U/2 249U/2 239U/2 235U/2 227U/2 223U/2 213U/2
                                            209U/2 201U/2 197U/2  176U   155U    1D
     MAIN.         ENTRY
    5 MAXH     I   VARIABLE       POINTEE    263P   251P   237P   225P   211P   199P   189N   177N   173P
                                             171N   168N   156N   144P   142N   139N   128N    24P
    7 MAXV     I   VARIABLE       POINTEE    269P   267N   257P   255N   243P   241N   231P   229N   217P
                                             215N   205P   203N   188N   175N   170N   167N   154N   141N
                                             138N   127N    25P
    4 MINH     I   VARIABLE       POINTEE   263U/3 251U/3 237U/3 225U/3 211U/3 199U/3  189N   177N  173U/2
                                             171N   168N   156N  144U/2  142N   139N   128N    24P
    6 MINV     I   VARIABLE       POINTEE   269U/3  267N  257U/3  255N  243U/3  241N  231U/3  229N  217U/3
```

					215N	205U/3	203N	188N	175N	170N	167N	154N	141N
					138N	127N	25P						
446	MPABS	I	VARIABLE		268U	265U/2	263U	261U/2	256U	253U/2	251U	249U/2	242U
					239U/2	237U	235U/2	230U	227U/2	225U	223U/2	216U	213U/2
					211U	209U/2	204U	201U/2	199U	197U/2	193S		
7	MPLOT	I	1DIM ARRAY	GRFCOM	260U	248U	234U	222U	208U	196U	193P	70U	40D
					35D								
0	MSLICE	I	VARIABLE	GRFCOM	194P	176P	155P	43P	40D				
	NDM1	I	PARAMETER		32P/3	31P/3	28P	27P/3	22P/2	13P/2	8P/3	7P	6P/3
					5S								
	NDM12	I	PARAMETER		20P/2	19S							
	NDM2	I	PARAMETER		28P	27P/3	22P/2	13P/2	8P/3	7P	6P/3	5S	
443	NH	I	VARIABLE		172U/3	171I	169U/8	168I	143U/3	142I	140U/8	139I	129U/8
					128I								
	NHV	I	PARAMETER		28P/2	26S							
4	NLABEL	I	VARIABLE	DUM.ARG.	83P	80P	41U	41P	1D				
	NNNX	I	PARAMETER		2S								
	NNNX2	I	PARAMETER		4S								
	NNNXP	I	PARAMETER		12P	11P	10P	3S					
	NNNY	I	PARAMETER		10P	2S							
	NNNY2	I	PARAMETER		12P	11P	4S						
	NNNYP	I	PARAMETER		3S								
	NNNZ	I	PARAMETER		10P	2S							
	NNNZ2	I	PARAMETER		15P/2	12P	11P	4S					
	NNNZP	I	PARAMETER		3S								
	NNYZ2	I	PARAMETER		18S								
	NOBRDR		EXTERNAL		73U								
	NSPEC	I	PARAMETER		273U	173P	144P	18S					
442	NV	I	VARIABLE		268U/2	267I	256U/2	255I	242U/2	241I	230U/2	229I	216U/2
					215I	204U/2	203I	173U/5	172U/2	170I	169U/8	167I	144U/5
					143U/2	141I	140U/8	138I	129U/8	127I			
	NVAR	I	PARAMETER		20P	18S							
	PAGE		EXTERNAL		72U								
132	PCAMB	R	1DIM ARRAY	HYDFIX	176P/2	16D	15D						
0	PRE	R	2D EQ. ARRAY	LOCOM1	178U	173P	144P	137P	29D	27D			
454	PROF3D		ENTRY		1D/2								
15114	PSCR	R	1D EQ. ARRAY	LOCOM1	178U	176S	173P	144P	33D	31D			
	PWRIT		EXTERNAL		266U	262U	240U	236U	214U	210U	84U	83U	
0	RCAMB	R	1DIM ARRAY	HYDFIX	155P/2	16D	15D						
2	RFSTF	CH	VARIABLE	DUM.ARG.	42U	37D	1D						
75700	RHA	R	2DIM ARRAY	PLTVAR	173P	144P	102P	9D	8D				
112200	RHB	R	2DIM ARRAY	PLTVAR	173P	144P	110P	9D	8D				
61400	RHO	R	2DIM ARRAY	PLTVAR	190U	173P	169U	157U	144P	140U	129U	94P	9D
					8D								
15004	RSCR	R	1D EQ. ARRAY	LOCOM1	173P	157U	155S	144P	33D	31D			
0	RVX	R	2DIM ARRAY	PLTVAR	169U/2	140U/2	129U/2	9D	6D				
14300	RVY	R	2DIM ARRAY	PLTVAR	169U/2	140U/2	129U/2	9D	6D				
30600	RVZ	R	2DIM ARRAY	PLTVAR	190U	169U/2	140U/2	129U/2	9D	6D			
15334	SCRA	R	1D EQ. ARRAY	LOCOM1	173P	144P	34D	32D					
15444	SCRB	R	1D EQ. ARRAY	LOCOM1	173P	144P	34D	32D					
15554	SCRC	R	1D EQ. ARRAY	LOCOM1	173P	144P	34D	32D					
101024	SCRH	R	1D EQ. ARRAY	LOCOM1	269P	268S	257P	256S	243P	242S	231P	230S	217P
					216S	205P	204S	30D	28D				
	SHDCHR		EXTERNAL		76U								
0	SPACE	R	2DIM ARRAY	HOLDER	257P	251P	231P	225P	205P	199P	13D		
14302	SPACE2	R	2DIM ARRAY	HOLDER	190S	187P	178S	166P	157S	153P	13D		

```
         SWISSM         EXTERNAL                      77U
       3 TIME      R    VARIABLE       GLOVAR         42U    17D
       3 VCOR      R    1DIM ARRAY     POINTEE        269P   261U   257P   249U   243P   235U   231P   223U   217P
                                                      209U   205P   197U   23P    22D
       0 XCOR      R    1DIM ARRAY     GRDVAR         65P    49P    14D    10D
         XNAME          EXTERNAL                      182U   161U   148U   132U   121U   113U   105U   97U    89U
     111 YCOR      R    1DIM ARRAY     GRDVAR         265U   261U   253U   249U   57P    50P    14D    10D
         YNAME          EXTERNAL                      183U   162U   149U   133U   122U   114U   106U   98U    90U
     152 ZCOR      R    1DIM ARRAY     GRDVAR         239U   235U   227U   223U   213U   209U   201U   197U   66P
                                                      58P    14D    10D
TABLE OF EXTERNAL NAMES
-----------------------

         $EFA           EXTERNAL                      42U    41U
         $EFF           EXTERNAL                      265U   261U   253U   249U   239U   235U   227U   223U   213U
                                                      209U   201U   197U   42U    41U
         $EFI           EXTERNAL                      265U   261U   253U   249U   239U   235U   227U   223U   213U
                                                      209U   201U   197U   42U    41U
         $EFV           EXTERNAL                      265U/4 261U/4 253U/4 249U/4 239U/4 235U/4 227U/4 223U/4 213U/4
                                                      209U/4 201U/4 197U/4 42U/2
         ANOTAT         EXTERNAL                      185U   164U   151U   135U   124U   116U   108U   100U   92U
         AREA2D         EXTERNAL                      74U
         DISIPL         EXTERNAL                      257U   251U   231U   225U   205U   199U
         EOS            EXTERNAL                      173U   144U
         FZXY           EXTERNAL                      269U   263U   243U   237U   217U   211U
         HEADIN         EXTERNAL                      254U   250U   228U   224U   202U   198U   81U    80U
         HEIGHT         EXTERNAL                      79U
         NOBRDR         EXTERNAL                      73U
         PAGE           EXTERNAL                      72U
         PWRIT          EXTERNAL                      266U   262U   240U   236U   214U   210U   84U    83U
         SHDCHR         EXTERNAL                      76U
         SWISSM         EXTERNAL                      77U
         XNAME          EXTERNAL                      182U   161U   148U   132U   121U   113U   105U   97U    89U
         YNAME          EXTERNAL                      183U   162U   149U   133U   122U   114U   106U   98U    90U
      ABBREVIATIONS USED ABOVE (THESE ARE KEYED TO THE SOURCE LISTING LINE NUMBER)
      ---------------------------------------

      A  USED IN FORTRAN  ASSIGN STATEMENT      P  USED IN CALL/FUNC CALL OR ARRAY DEF
      D  DEFINED IN  DECLARATIVE STATEMENT      R  FORMAT  USED  IN A  READ  STATEMENT
      E  STATEMENT NUMBER ENDING A DO LOOP      S  STORED  SO CONTENTS  MAY BE CHANGED
      I  INDEX OF A DO  OR IMPLIED DO LOOP      U  NAME  USED IN EXECUTABLE  STATEMENT
      J  STATEMENT NUMBER USED IN TRANSFER      W  FORMAT  USED IN A  WRITE  STATEMENT
      L  SOURCE LINE OF A STATEMENT NUMBER      *  DEFINED  OR DECLARED  BUT NOT  USED
      N  NAME USED AS A  DO LOOP PARAMETER      ?  TEN OR  MORE REFERENCES  TO  SYMBOL
TABLE OF PARAMETERS ENCOUNTERED
-------------------------------

NDM1      = 72                                    NDM12     = 6660
NDM2      = 88                                    NHV       = 88
NNNX      = 72                                    NNNX2     = 74
NNNXP     = 73                                    NNNY      = 33
NNNY2     = 35                                    NNNYP     = 34
NNNZ      = 88                                    NNNZ2     = 90
NNNZP     = 89                                    NNYZ2D    = 3150
```

NSPEC = 1 NVAR = 5

TABLE OF LOOPS ENCOUNTERED

LABEL	INDEX	FROM	TO	ADDRESS	LENGTH
240	IP	69	274	573a	2043
120	NV	127	129	1031d	34
120	NH	128	129	1042b	17
135	NV	138	140	1113d	34
135	NH	139	140	1124b	17
150	NV	141	145	1152c	64
140	NH	142	143	1164b	11
1002	JJV	154	158	1265b	32
1002	IIH	156	158	1304b	7
1011	NV	167	169	1345d	34
1011	NH	168	169	1356b	17
1013	NV	170	174	1404c	64
1012	NH	171	172	1416b	11
1015	JJV	175	179	1474a	31
1015	IIH	177	179	1513a	7
1022	JJV	188	191	1554c	25
1022	IIH	189	191	1565a	11
180	NV	203	204	1721b	6
185	NV	215	216	2050c	6
200	NV	229	230	2200b	6
205	NV	241	242	2327c	6
220	NV	255	256	2457b	6
225	NV	267	268	2606c	6

BLOCK NAMES AND LENGTHS IN OCTAL

2641-PROF3D	57 #IB	706 #CL	126500 PLTVAR	30602-HOLDER	1116-GRDVAR
264-HYDFIX	14 GLOVAR	116030 LOCOM1	35 GRFCOM		

STATIC SPACE (IN OCTAL)

B SAVE:	47
T SAVE:	3
CONSTANTS:	402
VARIABLES:	45
TEMPORARIES:	713
CODE:	2172
TOTAL:	3626

```
  1.          subroutine fileio (ifetch,nf)
        c   ------- ------ ------ ---
        c   performs disk reads using the CFILIB random access I/O routine
        c   RDABSF. checks for I/O completion before returning to calling
        c   program.
        c ------- array dimensions  --- ------------- ------
  2.          parameter (nnnx=72, nnny=33, nnnz=88)
  3.          parameter (nnyz2d=(nnny+2)*(nnnz+2))
        c-------------- ----- hydro arrays --------------- --- ---
  4.          parameter (nspec=1, nvar=4+nspec, npl=1)
  5.          parameter (jkvar=nvar*nnyz2d)
  6.          real dvar(nnyz2d,nvar,npl)
        c
  7.          common /hydvar/ dvar
  8.          real var(nnyz2d,nvar,npl)
  9.          pointer (ipvar,var)
        c------------------- local declarations -------- -- ------- ---
 10.          logical lfetch, lstore
 11.          character *8 file
        c
        c   set pointer.
 12.          ipvar=loc(dvar)
        c
        c   perform I/O.
 13.          call rdabsf (lunit,var(1,1,nf),jkvar,(ifetch-1)*jkvar)
 14.          return
        c
 15.          entry filefo(file,lun,nsiz)
        c---------------------------------------------- ---------
        c   entry FILEFO opens the disk file.
        c   the arguments:
        c    file - a real variable containing the hollerith string filename
        c    lun    logical unit number
        c    nsiz   file size (words)
        c--------------- ------ -- ------------- -------
 16.          lunit=lun
 17.          nsize=nsiz
        c
        c   open the disk file.
 18.          call assign (lunit,file,0)
 19.          call famwait (lunit,1)
 20.          call famsiz (lunit,nsize)
 21.          return
        c
 22.          entry filefc
        c   ------- ----------- ----------------- ---
        c   entry FILEFC closes the disk file after ensuring I/O completion.
        c---- ----------------------------------------------------------
        c   confirm that all previous disk i/o requests have been completed.
 23.          if (unit(lunit)) 10,30,40
        c
        c   close the file.
 24.    10    continue
 25.          call close (lunit)
 26.          return
        c
```

```
27.              entry fileid
          c
          c     entry FILEID destroys the scratch disk I/O file.
          c-------------------------------------------------------------
28               call drabsf (lunit,file,0,ierr)
29               write (6,70) ierr
30.              return
          c
31.              entry fileck
          c-------------------------------------------------------------
          c     entry FILECK waits for all pending I/O requests to be completed
          c     before returning to the calling program.
          c-------------------------------------------------------------
32.              if (unit(lunit)) 20,30,40
          c
33.        20    return
          c
          c     error  messages.
34.        30    write (6,50) lunit
35.              stop
          c
36.        40    write (6,60) lunit
37.              stop
          c
38         50 format (/' EOF or EOI condition present for unit ','lunit = ',i3)
39.        60 format (///' I/O error occured for unit lunit = ',i3)
40.        70 format (//1x,'ierr from subroutine DRABSF=',i3)
41.           end
```

```
TABLE OF STATEMENT NUMBERS (ALL ADDRESSES IN TABLES ARE IN OCTAL)

  NUMBER USE    SOURCE PROGRAM REFERENCES

    10 UNDEF**     24L      23L      23J
    20 UNDEF**     33L      32L      32J
    30 217c        34L      32J      23J
    40 230a        36L      32J      23J
    50 FN          38L      34W
    60 FN          39L      36W
    70 FN          40L      29W
  (SN-STATEMENT NUMBER, GSN-GENERATED STATEMENT NUMBER)
  (FN-FORMAT NUMBER, UNDEF**-UNDEFINED STATEMENT NUMBER)
TABLE OF NAMES ENCOUNTERED (ADDRESS FOR DUMMY ARGUMENT IS THE ARGUMENT NUMBER)

 ADDRESS NAME     TYPE MAIN USAGE   BLOCK        SOURCE PROGRAM REFERENCES

         $STOP         EXTERNAL                   37U     35U
         $WFF          EXTERNAL                   36U     34U     29U
         $WFI          EXTERNAL                   36U     34U     29U
         $WFV          EXTERNAL                   36U     34U     29U
         ASSIGN        EXTERNAL                   18U
         CLOSE         EXTERNAL                   25U
         DRABSF        EXTERNAL                   28U
       0 DVAR      R   3DIM ARRAY   HYDVAR        12P      7D      6D
         FAMSIZ        EXTERNAL                   20U
         FAMWAIT       EXTERNAL                   19U
       3 FILE      CH  VARIABLE     DUM.ARG.      28P     18P     15D     11D
     204 FILECK        ENTRY                      31D/2
     136 FILEEC        ENTRY                      22D/2
     155 FILEFD        ENTRY                      27D/2
     102 FILEFJ        ENTRY                      15D/2
      50 FILEIO        ENTRY                      31D     27D     22D     15D      1D/2
      46 IERR      I   VARIABLE                   29U     28P
       1 IFETCH    I   VARIABLE     DUM.ARG.      13U      1D
      41 IPVAR     *I  VARIABLE                   12S      9P
         JKVAR     I   PARAMETER                  13U     13P      5S
         LOC       I   INLINE FUNCT               12U
       4 LUN       I   VARIABLE     DUM.ARG.      16U     15D
      44 LUNIT     I   VARIABLE                   36U     34U     32P     28P     25P     23P     20P     19P     18P
                                                  16S     13P
         MAIN.         ENTRY
       2 NF        I   VARIABLE     DUM.ARG.      13U      1D
         NNNX      I   PARAMETER                   2S
         NNNY      I   PARAMETER                   2S
         NNNZ      I   PARAMETER                   2S
         NNYZ2D    I   PARAMETER                   8P      6P      3S
         NPL       I   PARAMETER                   8P      6P      4S
       5 NSIZ      I   VARIABLE     DUM.ARG.      17U     15D
      45 NSIZE     I   VARIABLE                   20P     17S
         NSPEC     I   PARAMETER                   4S
```

```
        NVAR       I  PARAMETER                    8P      6P      45
        RDABSI        EXTERNAL                    13U
        UNIT       R  EXTERNAL                    32U     23U
      1 VAR        R  3DIM ARRAY    POINTEE       13P      9P      8D
TABLE OF EXTERNAL NAMES
------------------------

        $STOP         EXTERNAL                    37U     35U
        $WFF          EXTERNAL                    36U     34U     29U
        $WFI          EXTERNAL                    36U     34U     29U
        $WFV          EXTERNAL                    36U     34U     29U
        ASSIGN        EXTERNAL                    18U
        CLOSE         EXTERNAL                    25U
        DRABSF        EXTERNAL                    28U
        FAMSIZ        EXTERNAL                    20U
        FAMWAIT       EXTERNAL                    19U
        RDABSF        EXTERNAL                    13U
        UNIT       R  EXTERNAL                    32U     23U
      ABBREVIATIONS USED ABOVE (THESE ARE KEYED TO THE SOURCE LISTING LINE NUMBER)
      --------------------------------
      A  USED IN FORTRAN  ASSIGN STATEMENT     P  USED IN CALL/FUNC CALL OR ARRAY DEF
      D  DEFINED IN  DECLARATIVE STATEMENT     R  FORMAT  USED  IN A  READ  STATEMENT
      E  STATEMENT NUMBER ENDING A DO LOOP     S  STORED  SO CONTENTS  MAY BE CHANGED
      I  INDEX OF A DO  OR IMPLIED DO LOOP     U  NAME  USED IN EXECUTABLE  STATEMENT
      J  STATEMENT NUMBER USED IN TRANSFER     W  FORMAT  USED IN A  WRITE  STATEMENT
      L  SOURCE LINE OF A STATEMENT NUMBER     *  DEFINED  OR DECLARED  BUT NOT  USED
      N  NAME USED AS A  DO LOOP PARAMETER     ?  TEN OR  MORE REFERENCES  TO  SYMBOL
TABLE OF PARAMETERS ENCOUNTERED
-------------------------------

 JKVAR     = 15750                                            NNNX     = 72
 NNNY      = 33                                               NNNZ     = 88
 NNYZ2D    = 3150                                             NPL      = 1
 NSPEC     = 1                                                NVAR     = 5

 BLOCK NAMES AND LENGTHS IN OCTAL

     241-FILEIO          5 #IB             77 #CL            36606 HYDVAR
STATIC SPACE (IN OCTAL)
-----------------------
B SAVE:               4
I SAVE:               0
CONSTANTS:           37
VARIABLES:           10
TEMPORARIES:        100
CODE:               172
               --------
TOTAL:              345
```

```
  1.           subroutine pltarr (ii,jj,kk,nloc)
        c  ----------------------------------------------------------
        c      fills plot arrays rvx, rvy, rvz, erg, rho, rha, rhb.
        c  ------------------ array dimensions ----------------------
  2.           parameter (nnnx=72, nnny=33, nnnz=88)
  3.           parameter (nnny2=nnny+2, nnnz2=nnnz+2)
  4.           parameter (nnyz2d=nnny2+nnnz2)
        c  ------------------ hydro arrays --------------------------
  5.           parameter (nspec=1, nvar=4+nspec, npl=1)
  6.           real var(nnyz2d,nvar,npl)
  7.           common /hydvar/ var
        c----------- fixed arrays of hydrodynamic data --------------
  8.           parameter (nsum=4+nspec)
  9.           real rcamb(nnnz2), pcamb(nnnz2), ecamb(nnnz2)
 10.           real sumpl(nsum,nnnx)
 11.           real rcmin(nnnz2), gravz(nnnz2), grapz(nnnz2), fsky(nnnz2)
        c
 12.           common /hydfix/ rcamb, pcamb, ecamb, sumpl, rcmin, gravz,
            1  grapz, sum4, shrink, fsky, ratio
        c  ------------------ plot variables ------------------------
 13.           parameter (ndm1=72, ndm2=88)
 14.           real rvx(ndm1,ndm2), rvy(ndm1,ndm2), rvz(ndm1,ndm2)
 15.           real erg(ndm1,ndm2)
 16.           real rho(ndm1,ndm2), rha(ndm1,ndm2), rhb(ndm1,ndm2)
 17.           common /pltvar/ rvx, rvy, rvz, erg, rho, rha, rhb
        c  ------------------ local declarations --------------------
 18.           logical lintp, colour, displa, darkhd
 19.           real mplot(14)
 20.           common /grfcom/ mslice, imin, imax, jmin, jmax, kmin, kmax,
            1  mplot, lintp, colour, dmin, dmax, displa, il, jl, darkhd
        c
        c      get data along the appropriate slice.
 21.           go to (10,90,170), mslice
        c
        c      X-Y slice.
 22.     10    jk=kk+nnny2+1
 23.           do 20 j=jmin,jmax
 24.           rvx(ii,j)=var(j+jk,1,nloc)
 25.           rvy(ii,j)=var(j+jk,2,nloc)
 26.           rvz(ii,j)=var(j+jk,3,nloc)
 27.     20    erg(ii,j)=var(j+jk,4,nloc)
 28.           go to (70,50,30), nspec
 29.     30    do 40 j=jmin,jmax
 30.     40    rhb(ii,j)=cvmgt(var(j+jk,7,nloc),0.0,var(j+jk,7,nloc).gt.
            1  1.e-6*var(j+jk,5,nloc))
 31.     50    do 60 j=jmin,jmax
 32.     60    rha(ii,j)=cvmgt(var(j+jk,6,nloc),0.0,var(j+jk,6,nloc).gt.
            1  1.e-6*var(j+jk,5,nloc))
 33.     70    do 80 j=jmin,jmax
 34.     80    rho(ii,j)=var(j+jk,5,nloc)
 35.           go to 250
        c
        c      Y-Z slice
 36.     90    do 100 k=kmin,kmax
 37.           jk=k*nnny2+1
 38.           do 100 j=jmin,jmax
```

```
  1574   39.          rvx(j,k) var(j+jk,1,nloc)
  1575   40.          rvy(j,k)-var(j+jk,2,nloc)
  1576   41.          rvz(j,k)-var(j+jk,3,nloc)
  1577   42.   100    erg(j,k) var(j+jk,4,nloc)
  1578   43           go to (150,130,110), nspec
  1579   44.   110    do 120 k kmin,kmax
  1580   45.          jk=k+nnny2+1
  1581   46           do 120 j jmin,jmax
  1582   47.   120    rhb(j,k)-cvmgt(var(j+jk,7,nloc),0.0,var(j+jk,7,nloc) gt
  1583            1   1.e 6*var(j+jk,5,nloc))
  1584   48.   130    do 140 k-kmin,kmax
  1585   49.          jk-k+nnny2+1
  1586   50.          do 140 j-jmin,jmax
  1587   51.   140    rha(j,k)-cvmgt(var(j+jk,6,nloc),0.0,var(j+jk,6,nloc) gt
  1588            1   1.e 6*var(j+jk,5,nloc))
  1589      c  140     rha(j,k)-cvmgt(var(j+jk,6,nloc),0.0,var(j+jk,6,nloc) ge.
  1590      c    1     dmin.and.var(j+jk,6,nloc).le.dmax)
  1591   52.   150    do 160 k-kmin,kmax
  1592   53.          jk=k+nnny2+1
  1593   54.          do 160 j-jmin,jmax
  1594   55.   160    rho(j,k) var(j+jk,5,nloc)
  1595   56.          go to 250
  1596      c
  1597      c     X-Z slice.
  1598   57.   170    jk=jj+1
  1599   58.          do 180 k-kmin,kmax
  1600   59.          jk1-jk+k+nnny2
  1601   60.          rvx(ii,k)-var(jk1,1,nloc)
  1602   61.          rvy(ii,k)-var(jk1,2,nloc)
  1603   62.          rvz(ii,k)-var(jk1,3,nloc)
  1604   63.   180    erg(ii,k)=var(jk1,4,nloc)
  1605   64.          go to (230,210,190), nspec
  1606   65.   190    do 200 k-kmin,kmax
  1607   66.   200    rhb(ii,k)-cvmgt(var(jk+k+nnny2,7,nloc),0.0,
  1608            1     var(jk+k+nnny2,7,nloc) gt.1.e-6*var(jk+k+nnny2,5,nloc))
  1609   67.   210    do 220 k-kmin,kmax
  1610   68.   220    rha(ii,k)-cvmgt(var(jk+k+nnny2,6,nloc),0.0,
  1611            1     var(jk+k+nnny2,6,nloc).gt.1.e-6*var(jk+k+nnny2,5,nloc))
  1612      c  220     rha(ii,k)=cvmgt(var(jk+k+nnny2,6,nloc),0.0,
  1613      c    1     var(jk+k+nnny2,6,nloc).ge.dmin.and.var(jk+k+nnny2,6,nloc)
  1614      c    2     .le.dmax)
  1615   69.   230    do 240 k kmin,kmax
  1616   70.   240    rho(ii,k)-var(jk+k+nnny2,5,nloc)
  1617   71.   250    continue
  1618   72.          return
  1619   73.        end
PLTARR           VECTOR LOOP BEGINS AT SEQ. NO.      23, P=     27b
PLTARR           VECTOR LOOP BEGINS AT SEQ. NO.      29, P-     75c
PLTARR           VECTOR LOOP BEGINS AT SEQ. NO.      31, P=    123b
PLTARR           VECTOR LOOP BEGINS AT SEQ. NO       33, P=    151a
PLTARR           VECTOR LOOP BEGINS AT SEQ. NO.      38, P=    175c
PLTARR           VECTOR LOOP BEGINS AT SEQ. NO.      46, P=    247c
PLTARR           VECTOR LOOP BEGINS AT SEQ. NO.      50, P=    304c
PLTARR           VECTOR LOOP BEGINS AT SEQ. NO.      54, P=    341c
PLTARR           VECTOR LOOP BEGINS AT SEQ. NO.      65, P-    422d
PLTARR           VECTOR LOOP BEGINS AT SEQ. NO.      67, P-    451c
```

PLTARR VECTOR LOOP BEGINS AT SEQ. NO. 69, P- 500b

TABLE OF STATEMENT NUMBERS (ALL ADDRESSES IN TABLES ARE IN OCTAL)

```
 NUMBER USE    SOURCE PROGRAM REFERENCES

   10 27b         22L      21J
   20 UNDEF**     27L      23F
   20A43c         23L
   20B70a         23L
   30 75c         29L      28J
   40 UNDEF**     30L      29F
   40A110b        29L
   40B123b        29L
   50 123b        31L      28J
   60 UNDEF**     32L      31E
   60A136a        31L
   60B151a        31L
   70 151a        33L      28J
   80 UNDEF**     34L      33E
   80A162c        33L
   80B172a        33L
   90 172c        36L      21J
  100 UNDEF**     42L      38E      36F
  100A175c        36L
  100B237a        36L
  100C211d        38L
  100D234b        38L
  110 244c        44L      43J
  120 UNDEF**     47L      46F      44F
  120A247c        44L
  120B301c        44L
  120C265a        46L
  120D276d        46L
  130 301c        48L      43J
  140 UNDEF**     51L      50F      48F
  140A304c        48L
  140B336c        48L
  140C322a        50L
  140D333d        50L
  150 336c        52L      43J
  160 UNDEF**     55L      54F      52F
  160A341c        52L
  160B370b        52L
  160C356c        54L
  160D365c        54L
  170 370d        57L      21J
  180 UNDEF**     63L      58F
  180A403c        58L
  180B415b        58L
  190 422d        65L      64J
  200 UNDEF**     66L      65F
  200A435c        65L
  200B451c        65L
  210 451c        67L      64J
  220 UNDEF**     68L      67F
```

```
   220A464b      67I
   220B500b      67L
   230  500b     69I     64J
   240 UNDEF**   70I     69F
   240A511d      69L
   240B521d      69L
   250  521d     71L     56J     35J
 00002  25d      21W
 00003  74a      28W
 00004 243a      43W
 00005 421b      64W
  (SN=STATEMENT NUMBER, GSN=GENERATED STATEMENT NUMBER)
  (FN=FORMAT NUMBER, UNDEF**=UNDEFINED STATEMENT NUMBER)
TABLE OF NAMES ENCOUNTERED (ADDRESS FOR DUMMY ARGUMENT IS THE ARGUMENT NUMBER)
--------------------------

ADDRESS NAME     TYPE MAIN USAGE  BLOCK      SOURCE PROGRAM REFERENCES

      CVMGT           INLINE FUNCT.           68U    66U    51U    47U    32U    30U
45100 ERG        *R   2DIM ARRAY   PLTVAR     63S    42S    27S    17D    15D
    1 II          I   VARIABLE     DUM.ARG.   70U    68U    66U    63U    62U    61U    60U    34U    32U
                                              30U    27U    26U    25U    24U    1D
    5 J           I   VARIABLE                55U/2  54I    51U/4  50I    47U/4  46I    42U/2  41U/2  40U/2
                                              39U/2  38I    34U/2  33I    32U/4  31I    30U/4  29I    27U/2
                                              26U/2  25U/2  24U/2  23I
    2 JJ          I   VARIABLE     DUM.ARG.   57U    1D
    4 JK          I   VARIABLE                70U    68U/3  66U/3  59U    57S    55U    53S    51U/3  43S
                                              47U/3  45S    42U    41U    40U    39U    37S    34U    32U/3
                                              30U/3  27U    26U    25U    24U    22S
    7 JK1         I   VARIABLE                63U    62U    61U    60U    59S
    4 JMAX        I   VARIABLE     GRFCOM     54N    50N    46N    38N    33N    31N    29N    23N    20D
    3 JMIN        I   VARIABLE     GRFCOM     54N    50N    46N    38N    33N    31N    29N    23N    20D
    6 K           I   VARIABLE                70U/2  69I    68U/4  67I    66U/4  65I    63U    62U    61U
                                              60U    59U    58I    55U    53U    52I    51U    49U    48I
                                              47U    45U    44I    42U    41U    40U    39U    37U    36I
    3 KK          I   VARIABLE     DUM.ARG.   22U    1D
    6 KMAX        I   VARIABLE     GRFCOM     69N    67N    65N    58N    52N    48N    44N    36N    20D
    5 KMIN        I   VARIABLE     GRFCOM     69N    67N    65N    58N    52N    48N    44N    36N    20D
      MAIN.           ENTRY
    0 MSLICE      I   VARIABLE     GRFCOM     21P    20D
      NDM1        I   PARAMETER               16P/3  15P    14P/3  13S
      NDM2        I   PARAMETER               16P/3  15P    14P/3  13S
    4 NLOC        I   VARIABLE     DUM ARG.   70U    68U/3  66U/3  63U    62U    61U    60U    55U    51U/3
                                              47U/3  42U    41U    40U    39U    34U    32U/3  30U/3  27U
                                              26U    25U    24U    1D
      NNNX        I   PARAMETER               10P    2S
      NNNY        I   PARAMETER               2S
      NNNY2       I   PARAMETER               70U    68U/3  66U/3  59U    53U    49U    45U    37U    22U
                                              3S
      NNNZ        I   PARAMETER               2S
      NNNZ2       I   PARAMETER               11P/4  9P/3   3S
      NNYZ2D      I   PARAMETER               6P     4S
      NPL         I   PARAMETER               6P     5S
      NSPEC       I   PARAMETER               64P    43P    28P    5S
```

```
      NSUM     I  PARAMETER                  10P    8S
      NVAR     I  PARAMETER                  6P     5S
   15 PLTARR      ENTRY                      10/2
75700 RHA     *R  2DIM ARRAY   PLTVAR        68S    51S    32S    17D    16D
112200 RHB    *R  2DIM ARRAY   PLTVAR        66S    47S    30S    17D    16D
61400 RHD     *R  2DIM ARRAY   PLTVAR        70S    55S    34S    17D    16D
    0 RVX     *R  2DIM ARRAY   PLTVAR        60S    39S    24S    17D    14D
14300 RVY     *R  2DIM ARRAY   PLTVAR        61S    40S    25S    17D    14D
30600 RVZ     *R  2DIM ARRAY   PLTVAR        62S    41S    26S    17D    14D
    0 VAR      R  3DIM ARRAY   HYDVAR        70U    68U    68P/2  66U    66P/2  63U  62U  61U  60U
                                             55U    51U    51P/2  47U    47P/2  42U  41U  40U  39U
                                             34U    32U    32P/2  30U    30P/2  27U  26U  25U  24U
                                             7D     6D
```

TABLE OF EXTERNAL NAMES

ABBREVIATIONS USED ABOVE (THESE ARE KEYED TO THE SOURCE LISTING LINE NUMBER)

```
A  USED IN FORTRAN  ASSIGN STATEMENT      P  USED IN CALL/FUNC CALL OR ARRAY DEF
D  DEFINED IN  DECLARATIVE STATEMENT      R  FORMAT  USED  IN A  READ  STATEMENT
E  STATEMENT NUMBER ENDING A DO LOOP      S  STORED  SO CONTENTS  MAY BE CHANGED
I  INDEX OF A DO  OR IMPLIED DO LOOP      U  NAME  USED IN EXECUTABLE  STATEMENT
J  STATEMENT NUMBER USED IN TRANSFER      W  FORMAT  USED IN A  WRITE  STATEMENT
L  SOURCE LINE OF A STATEMENT NUMBER      *  DEFINED  OR DECLARED  BUT NOT  USED
N  NAME USED AS A  DO LOOP PARAMETER      ?  TEN OR  MORE REFERENCES  TO  SYMBOL
```

TABLE OF PARAMETERS ENCOUNTERED

```
NDM1    = 72                                  NDM2     = 88
NNNX    = 72                                  NNNY     = 33
NNNY2   = 35                                  NNNZ     = 88
NNNZ2   = 90                                  NNYZ2D   = 3150
NPL     = 1                                   NSPEC    = 1
NSUM    = 5                                   NVAR     = 5
```

TABLE OF LOOPS ENCOUNTERED

LABEL	INDEX	FROM	TO	ADDRESS	LENGTH
20	J	23	27	43c	25
40	J	21	30	110b	13
60	J	31	32	136a	13
80	J	33	34	162c	10
100	K	36	42	175c	42
100	J	38	42	211d	23
120	K	44	47	247c	33
120	J	46	47	265a	11
140	K	48	51	304c	33
140	J	50	51	322a	11
160	K	52	55	341c	27

```
   160 J            54    55    356c      7
   180 K            58    63    403c     12
   200 K            65    66    435c     14
   220 K            67    68    464b     14
   240 K            69    70    511d     10
 BLOCK NAMES AND LENGTHS IN OCTAL

    525-PLTARR            42-#IB         0-#CL         36606-HYDVAR       1741-HYDFIX      126500-PLTVAR
     35-GRFCOM
 STATIC SPACE (IN OCTAL)

 B SAVE:                41
 T SAVE:                 1
 CONSTANTS:              2
 VARIABLES:              6
 TEMPORARIES:            0
 CODE:                 515

 TOTAL:                567
```

```
  1.          subroutine restrt (istep,nmore,restf,label)
       c-----------------------------------------------------------------
       c    read data from dump file for graphics.
       c------------------------- array dimensions ----------------------
  2.          parameter (nnnx=72, nnny=33, nnnz=88)
  3.          parameter (nnnxp=nnnx+1, nnnyp=nnny+1, nnnzp=nnnz+1)
  4.          parameter (nnnx2=nnnx+2, nnny2=nnny+2, nnnz2=nnnz+2)
  5.          parameter (nnyz2d=nnny2*nnnz2)
       c----------------- hydro arrays ----------------------------------
  6.          parameter (nspec=1, nvar=4+nspec, npl=1)
  7.          parameter (jkpl=nnnx*nvar*nnyz2d, jkall=nnnx2*nvar*nnyz2d)
  8.          real dvar(nnyz2d,nvar,npl)
  9.          common /hydvar/ dvar
 10.          real var(nnyz2d,nvar,npl)
 11           pointer (ipvar,var)
       c----------- fixed arrays of hydrodynamic data -------------------
 12.          parameter (nsum=4+nspec, nfix=7*nnnz2+nnnx*nsum+3)
 13.          real rcamb(nnnz2), pcamb(nnnz2), ecamb(nnnz2)
 14.          real sumpl(nsum,nnnx)
 15.          real rcmin(nnnz2), gravz(nnnz2), grapz(nnnz2), fsky(nnnz2)
       c
 16.          common /hydfix/ rcamb, pcamb, ecamb, sumpl, rcmin, gravz,
          1   grapz, sum4, shrink, fsky, ratio
       c-------------- grid arrays ---------------------------------------
 17.          parameter (ngrd=2*(nnnxp+nnnyp+nnnzp))
 18.          real xcor(nnnxp), ycor(nnny), zcor(nnnz)
 19           real twodx(nnnxp), twody2(nnny2), twodz2(nnnz2)
 20.          real dtodx(nnnxp), dtody2(nnny2), dtodz2(nnnz2)
       c
 21.          common /grdvar/ xcor, ycor, zcor, twodx, twody2, twodz2, dtodx
          1   , dtody2, dtodz2
       c-------------------- global variables ----------------------------
 22.          parameter (ntim=9)
 23.          common /glovar/ dtmin, dtmax, cour, time, dt, dth, rlos, nfile
          1   , nstep, lllx, llly, lllz
       c---------------- body indices -----------------------------------
 24.          parameter (nnbd=1, nbodx=1, nbodz=1)
 25.          parameter (nbody=1+2*nnbd*(1+nbodx*(1+nbodz)))
       c        integer imnbd(nnbd), imxbd(nnbd)
       c        integer kmnbd(nbodx,nnbd), kmxbd(nbodx,nnbd)
       c        integer jmnbd(nbodz,nbodx,nnbd), jmxbd(nbodz,nbodx,nnbd)
       cc
       c        common /bodcom/ nbod, imnbd, imxbd, kmnbd, kmxbd, jmnbd, jmxb
       c------------------ station arrays --------------------------------
 26.          parameter (mxx=1, nsta=1)
 27.          parameter (nstat=1+4*mxx*nsta+mxx+3*nsta+1)
 28.          logical lstat
 29.          real rhs(mxx,nsta), vls(mxx,nsta), gms(mxx,nsta)
 30.          real prs(mxx,nsta), tme(mxx), xs(nsta), ys(nsta), zs(nsta)
 31.          common /stats/ lstat, rhs, vls, gms, prs, tme, xs, ys, zs, nxx
       c
       c-------------------------------particles--------------------------
 32.          parameter (nop=70,nburst=2)
 33.          parameter (npartl=3*nop*nburst+nburst)
 34.          real xp(nop,nburst), zp(nop,nburst)
 35.          integer nopp(nburst),nchk(nop,nburst)
```

```
36.        common /partl/ xp,zp,nopp,nchk
      c
      c                     active grid boundaries ------ --------- ----
37.        parameter (nscn=6, nact=17)
38.        logical lscan, lsides, ltop, botm, lrear, lfront
39.        common /scan/ lscan, lsides, ltop, botm, lrear, lfront
40.        common /active/ ixr, ixl, ixro, ixlo, jyp, jya, jypo, jyao,
         1   kzt, kzb, kzto, kzbo, jks1, jks2, jkd1, jkd2, ktherm
      c                  local declarations --------------- -- ----
41.        parameter (nlab=5, nch=8*nlab)
42.        character *8 restf
43.        character *40 label
44         logical incore
      c
45.        parameter (nsize=jkall+ntim+nlab+nscn+nact+ngrd+nfix+nbody
         1   +nstat+npartl)
46.        data lunit /16/, ldump /17/
      c
      c    set pointer
47.            ipvar=loc(dvar)
      c
      c    open dump file.
48.            call ffleto (restf,lunit,nsize)
      c
      c    first read all the data stored at the end of the file. the
      c    conserved variables on the nnnx2 planes are stored at the
      c    beginning
      c    read file number, step, time, timestep and label.
49.            call rdabsf (lunit,time,ntim,jkall)
50.            call rdabsf (lunit,label,nlab,jkall+ntim)
51.            call fileck
      c
      c    write messages.
52.            write (6,30) nfile,nstep,time
53.            istep=nstep
      c
      c    error exits.
54.            if (lllx.ne.nnnx.or.llly.ne.nnny.or.lllz.ne.nnnz) then
55.            write (6,40) lllx,llly,lllz,nnnx,nnny,nnnz
56.            stop
57.            endif
      c
      c    check if this is a time history run.
58             if (lstat) then
59             iadrs=jkall+ntim+nlab+nscn+nact+ngrd
60.            go to 10
61.            endif
      c
      c    read active grid switches and boundary indices.
62.            call rdabsf (lunit,lscan,nscn,jkall+ntim+nlab)
63.            call rdabsf (lunit,ixr,nact,jkall+ntim+nlab+nscn)
      c
      c    read coordinate data.
64.            call rdabsf (lunit,xcor,ngrd,jkall+ntim+nlab+nscn+nact)
65.            iadrs=jkall+ntim+nlab+nscn+nact+ngrd
      c
```

```
              c      read fixed arrays of hydrodynamic data.
    66.                 call rdabsf (lunit,rcamb,nfix,iadrs)
              c
              c      read station arrays.
    67.          10     call rdabsf (lunit,lstat,nstat,iadrs+nfix)
              c
              c      read body indices.
              c          call rdabsf (lunit,nbod,nbody,iadrs+nfix+nstat)
              c
              c
              c     read particles
              c
    68.                 call rdabsf (lunit,xp,npartl,iadrs+nfix+nstat+nbody)
              c
    69.                 return
              c
    70.                 entry readin
              c-------------------------------------------------------------
              c      read in all the planes for incore calculation.
              c-------------------------------------------------------------
    71.                 call rdabsf (lunit,var,jkpl,0)
    72.                 call filefc
    73.                 return
              c
    74.                 entry datain(imin,imax,jj,kk)
              c-------------------------------------------------------------
              c      fill the plot arrays.
              c-------------------------------------------------------------
    75.                 do 20 ii=imin,imax
    76.                 if ( not.incore) then
    77.                 call fileck
    78.                 call fileio (ii,1)
    79.                 call fileck
    80.                 endif
    81.                 iloc=cvmgt(ii,1,incore)
    82.                 call pltarr (ii,jj,kk,iloc)
    83.          20     continue
    84.                 return
              c
    85.          30 format (1x,'reading file',i3,' at time, step=',e12.5,i6)
    86.          40 format ('***error***  wrong restart file'/1x,'dump file contains n
                  1nnx,nnny,nnnz=',3i4/1x,'fast3d expects nnnx,nnny,nnnz=',3i4/1x,'ex
                  2ecution terminated')
    87.            end
```

```
TABLE OF STATEMENT NUMBERS (ALL ADDRESSES IN TABLES ARE IN OCTAL)

  NUMBER USE     SOURCE PROGRAM REFERENCES

     10 227c       67L     60J
     20 UNDEF**    83L     75E
     20A312c       75L
     20B331d       75L
     30 FN         85I     52W
     40 FN         86L     55W
  00002 167c       54I
  00003 174d       58I
  00004 320b       76I
   (SN=STATEMENT NUMBER, GSN=GENERATED STATEMENT NUMBER)
   (FN=FORMAT NUMBER, UNDEF**=UNDEFINED STATEMENT NUMBER)
TABLE OF NAMES ENCOUNTERED (ADDRESS FOR DUMMY ARGUMENT IS THE ARGUMENT NUMBER)

 ADDRESS NAME     TYPE MAIN USAGE   BLOCK        SOURCE PROGRAM REFERENCES

         $STOP         EXTERNAL                    56U
         $WFF          EXTERNAL                    55U      52U
         $WFI          EXTERNAL                    55U      52U
         $WFV          EXTERNAL                    55U/6    52U/3
         CVMGT         INLINE FUNCT.               81U
     271 DATAIN        ENTRY                       74D/2
       0 DVAR      R   3DIM ARRAY    HYDVAR        47P       9D      8D
         FILECK        EXTERNAL                    79U      77U     51U
         FILEFC        EXTERNAL                    72U
         FILEFO        EXTERNAL                    48U
         FILEIO        EXTERNAL                    78U
      60 IADRS     I   VARIABLE                    68P      67P     66P     65S     59S
      61 II        I   VARIABLE                    82P      81P     78P     75I
      62 ILOC      I   VARIABLE                    82P      81S
       6 IMAX      I   VARIABLE      DUM.ARG.      75N      74D
       5 IMIN      I   VARIABLE      DUM.ARG.      75N      74D
       2 INCORE    I   VARIABLE      DUM.ARG.      81P      76U     44D      1D
      55 IPVAR    *I   VARIABLE                    47S      11P
       1 ISTEP    *I   VARIABLE      DUM.ARG.      53S       1D
       0 IXR       I   VARIABLE      ACTIVE        63P      40D
       7 JJ        I   VARIABLE      DUM.ARG.      82P      74D
         JKALL     I   PARAMETER                   65U      64P     63P     62P     59U     50P     49P     7S
         JKPL      I   PARAMETER                   71P       7S
      10 KK        I   VARIABLE      DUM.ARG.      82P      74D
       4 LABEL    CH   VARIABLE      DUM.ARG.      50P      43D      1D
      57 LDUMP    *I   VARIABLE                    46S
      11 LLLX      I   VARIABLE      GLOVAR        55U      54U     23D
      12 LLLY      I   VARIABLE      GLOVAR        55U      54U     23D
      13 LLLZ      I   VARIABLE      GLOVAR        55U      54U     23D
         LOC       I   INLINE FUNCT                47U
       0 LSCAN     L   VARIABLE      SCAN          62P      39D     38D
       0 LSTAT     L   VARIABLE      STATS         67P      58U     31D     28D
```

```
   56 LUNIT   I   VARIABLE                71P    68P    67P    66P    64P    63P    62P    50P    49P
                                          48P    46S
      MAIN.       ENTRY
      MXX     I   PARAMETER               30P/2  29P/3  26S
      NACT    I   PARAMETER               65U    64U    63P    59U    37S
      NBODX   I   PARAMETER               24S
      NBODY   I   PARAMETER               68U    25S
      NBODZ   I   PARAMETER               24S
      NBURST  I   PARAMETER               35P/2  34P/2  32S
      NCH     I   PARAMETER               41S
    7 NFILE   I   VARIABLE     GLOVAR     52U    23D
      NFIX    I   PARAMETER               68U    67U    66P    12S
      NGRD    I   PARAMETER               65U    64P    59U    17S
      NLAB    I   PARAMETER               65U    64U    63U    62U    59U    50P    41S
      NNBD    I   PARAMETER               24S
      NNNX    I   PARAMETER               55U    54U    14P    2S
      NNNX2   I   PARAMETER               4S
      NNNXP   I   PARAMETER               20P    19P    18P    3S
      NNNY    I   PARAMETER               55U    54U    18P    2S
      NNNY2   I   PARAMETEP               20P    19P    4S
      NNNYP   I   PARAMETER               3S
      NNNZ    I   PARAMETER               55U    54U    13P    2S
      NNNZ2   I   PARAMETER               20P    19P    15P/4  13P/3  4S
      NNNZP   I   PARAMETER               3S
      NNYZ2D  I   PARAMETER               10P    8P     5S
      NOP     I   PARAMETER               35P    34P/2  32S
      NPARTL  I   PARAMETER               68P    33S
      NPL     I   PARAMETER               10P    8P     6S
      NSCN    I   PARAMETER               65U    64U    63U    62P    59U    37S
      NSIZE   I   PARAMETER               48P    45S
      NSPEC   I   PARAMETER               6S
      NSTA    I   PARAMETER               30P/4  29P/3  26S
      NSTAT   I   PARAMETER               68U    67P    27S
   10 NSTEP   I   VARIABLE     GLOVAR     53U    52U    23D
      NSUM    I   PARAMETER               14P    12S
      NTIM    I   PARAMETER               65U    64U    63U    62U    59U    50U    49P    22S
      NVAR    I   PARAMETER               10P    8P     6S
      PLTARR      EXTERNAL                82U
    0 RCAMB   R   1DIM ARRAY   HYDFIX     66P    16D    13D
      RDABSF      EXTERNAL                71U    68U    67U    66U    64U    63U    62U    50U    49U
  252 READIN      ENTRY                   70D/2
    3 RESTF   CH  VARIABLE     DUM.ARG.   48P    42D    1D
   64 RESTRT      ENTRY                   74D    70D    1D/2
    3 TIME    R   VARIABLE     GLOVAR     52U    49P    23D
    1 VAR     R   3DIM ARRAY   POINTEE    71P    11P    10D
    0 XCOR    R   1DIM ARRAY   GRDVAR     64P    21D    18D
    0 XP      R   2DIM ARRAY   PARTL      68P    36D    34D
TABLE OF EXTERNAL NAMES
--- -- -------- -----

      $STOP       EXTERNAL                56U
      $WFF        EXTERNAL                55U    52U
      $WFI        EXTERNAL                55U    52U
      $WFV        EXTERNAL                55U/6  52U/3
      FILECK      EXTERNAL                79U    77U    51U
```

```
      FILEFC      EXTERNAL                    72U
      FILEFO      EXTERNAL                    48U
      FILEIO      EXTERNAL                    78U
      PLTARR      EXTERNAL                    82U
      RDABST      EXTERNAL                    71U    68U    67U    66U    64U    63U    62U    50U    49U
    ABBREVIATIONS USED ABOVE (THESE ARE KEYED TO THE SOURCE LISTING LINE NUMBER)
    ------------------------------------
    A  USED IN FORTRAN  ASSIGN STATEMENT      P  USED IN CALL/FUNC CALL OR ARRAY DEF
    D  DEFINED IN  DECLARATIVE STATEMENT      R  FORMAT  USED  IN A  READ  STATEMENT
    E  STATEMENT NUMBER ENDING A DO LOOP      S  STORED  SO CONTENTS  MAY BE CHANGED
    I  INDEX OF A DO  OR IMPLIED DO LOOP      U  NAME  USED IN EXECUTABLE  STATEMENT
    J  STATEMENT NUMBER USED IN TRANSFER      W  FORMAT  USED IN A  WRITE  STATEMENT
    L  SOURCE LINE OF A STATEMENT NUMBER      *  DEFINED  OR DECLARED  BUT NOT  USED
    N  NAME USED AS A  DO LOOP PARAMETER      ?  TEN OR  MORE REFERENCES  TO  SYMBOL
 TABLE OF PARAMETERS ENCOUNTERED
 ------------------------------

  JKALL    = 1165500                                     JKPL     = 1134000
  MXX      = 1                                           NACT     = 17
  NBODX    = 1                                           NBODY    = 7
  NBODZ    = 1                                           NBURST   = 2
  NCH      = 40                                          NFIX     = 993
  NGPD     = 392                                         NLAB     = 5
  NNBD     = 1                                           NNNX     = 72
  NNNX2    = 74                                          NNNXP    = 73
  NNNY     = 33                                          NNNY2    = 35
  NNNYP    = 34                                          NNNZ     = 88
  NNNZ2    = 90                                          NNNZP    = 89
  NNYZ2D   = 3150                                        NOP      = 70
  NPARTL   = 422                                         NPL      = 1
  NSCN     = 6                                           NSIZE    = 1167361
  NSPEC    = 1                                           NSTA     = 1
  NSTAT    = 10                                          NSUM     = 5
  NTIM     = 9                                           NVAR     = 5

 TABLE OF LOOPS ENCOUNTERED
 --------------------------
  LABEL INDEX      FROM    TO   ADDRESS - LENGTH

    20 II            75    83      312c        20
  BLOCK NAMES AND LENGTHS IN OCTAL

     335-RESTRT          24-#IB            141 #CL         36606-HYDVAR        1741-HYDFIX        1116-GRDVAR
      14-GLOVAR          12-STATS          646 PARTL           6-SCAN            21-ACTIVE
 STATIC SPACE (IN OCTAL)
 -----------------------
 B SAVE:             10
 T SAVE:              4
 CONSTANTS:          53
```

VARIABLES:	10
TEMPORARIES:	151
CODE	252

TOTAL:	522

```
1776      1.          subroutine colcon (arr,imin,jmin,kh,kv)
1777           c-----------------------------------------------------------------
1778           c      produces color contour plots
1779           c      this routine divides the contours into five ranges, and draws them
1780           c      in different colors according to which range their values fall in.
1781           c-----------------------------------------------------------------
1782      2.          parameter (ndm1=72, ndm2=88, nr=5)
1783      3           real arr(ndm1,ndm2), hue(nr)
1784      4.          common /holder/ space(ndm1,ndm2),valmin,valrng
1785      5.          logical displa, darkhd
1786      6.          common /grfcom/ mslice, iamin, imax, jamin, jmax, kmin, kmax,
1787              1   mplot, lintp, colour, dmin, dmax, displa, il, jl, darkhd
1788      7.          data hue /3hred,4hblue,5hgreen,6hyellow,5hclear/
1789           c
1790           c      determine maximum and minimum values.
1791      8.          arrmax=0.0
1792      9           arrmin=1.e50
1793     10.          do 10 j=jmin,jmin+kv-1
1794     11.          imx=ismax(kh,arr(imin,j),1)+imin-1
1795     12.          arrmax=amax1(arrmax,arr(imx,j))
1796     13.          imn=ismin(kh,arr(imin,j),1)+imin-1
1797     14.          arrmin=amin1(arrmin,arr(imn,j))
1798     15.    10    continue
1799     16.          drange=(arrmax-arrmin)/float(nr)
1800           c
1801           c      plot contours for each range.
1802           c      DISSPLA plots the colour contours in the subroutine discol.  NCAR
1803           c      draws the contour lines a few at a time, changing color between.
1804     17.          if (displa) then
1805     18.            valmin=arrmin
1806     19.            valrng=drange
1807     20.            call setclr ('RED')
1808     21.            call discol (arr(imin,jmin),ndm1,space,kh,kv)
1809     22.          else
1810     23.            do 30 ir=1,nr
1811     24.            flo=arrmin+float(ir-1)*drange
1812     25.            fhi=arrmin+float(ir)*drange
1813     26.            call flush
1814     27.            call gcolor (hue(ir))
1815     28.            call conrec (arr(imin,jmin),ndm1,kh,kv,flo,fhi,6,-1,0,0)
1816     29.            call flush
1817     30.    30      continue
1818     31.          endif
1819     32.          return
1820     33.        end
O  AT SEQUENCE NUMBER -      11.
 PRNAME COLCON   COMMENT -      DEPENDENCY INVOLVING ARRAY "ARR" IN SEQUENCE NUMBER 12        ****************************** P=02506300
 EXPLANATION:  ARRAY USED AS AN ARGUMENT TO A SUBROUTINE/FUNCTION
O  AT SEQUENCE NUMBER -      11.
 PRNAME COLCON   COMMENT -      DEPENDENCY INVOLVING ARRAY "ARR" IN SEQUENCE NUMBER 13        ****************************** P=02506300
 EXPLANATION:  ARRAY USED AS AN ARGUMENT TO A SUBROUTINE/FUNCTION
```

```
 TABLE OF STATEMENT NUMBERS (ALL ADDRESSES IN TABLES ARE IN OCTAL)

   NUMBER USE    SOURCE PROGRAM REFERENCES

     10 UNDEF**      15I       10E
     10A50c          10I
     10B77a          10I
     30 UNDEF**      30I       23L
     30A132d         23L
     30BUNDEF**      23L
  00002 121c         17I
  00003 161a         22I
  (SN=STATEMENT NUMBER, GSN GENERATED STATEMENT NUMBER)
  (FN=FORMAT NUMBER, UNDEF**-UNDEFINED STATEMENT NUMBER)
 TABLE OF NAMES ENCOUNTERED (ADDRESS FOR DUMMY ARGUMENT IS THE ARGUMENT NUMBER)

  ADDRESS NAME     TYPE MAIN USAGE   BLOCK         SOURCE PROGRAM REFERENCES

          AMAX1     R   INLINE FUNCT.                 12U
          AMIN1     R   INLINE FUNCT.                 14U
        1 ARR       R   2DIM ARRAY   DUM.ARG.         28P    21P    14P    13P    12P    11P    3D    1D
       17 ARRMAX    R   VARIABLE                      16U    12P    12S    8S
       20 ARRMIN    R   VARIABLE                      25U    24U    18U    16U    14P    14S    9S
       31 COLCON        ENTRY                          1D/2
          CONREC        EXTERNAL                      28U
          DISCOL        EXTERNAL                      21U
       14 DISPLA    L   VARIABLE     GRFCOM           17U    6D     5D
       24 DRANGE    R   VARIABLE                      25U    24U    19U    16S
       27 FHI       R   VARIABLE                      28P    25S
       26 FLO       R   VARIABLE                      28P    24S
          FLOAT     R   INLINE FUNCT                  25U    24U    16U
          FLUSH         EXTERNAL                      29U    26U
          GCOLOR        EXTERNAL                      27U
       12 HUE       R   1DIM ARRAY                    27P    7S     3D
        2 IMIN      I   VARIABLE     DUM.ARG.         28U    21U    13U/2  11U/2  1D
       23 IMN       I   VARIABLE                      14U    13S
       22 IMX       I   VARIABLE                      12U    11S
       25 IR        I   VARIABLE                      27U    25P    24P    23I
          ISMAX     I   EXTERNAL                      11U
          ISMIN     I   EXTERNAL                      13U
       21 J         I   VARIABLE                      14U    13U    12U    11U    10I
        3 JMIN      I   VARIABLE     DUM.ARG.         28U    21U    10N/2  1D
        4 KH        I   VARIABLE     DUM.ARG.         28P    21P    13P    11P    1D
        5 KV        I   VARIABLE     DUM.ARG.         28P    21P    10N    1D
          MAIN.         ENTRY
          NDM1      I   PARAMETER                     28P    21P    4P     3P     2S
          NDM2      I   PARAMETER                     4P     3P     2S
          NR        I   PARAMETER                     23N    16P    3P     2S
          SETCLR        EXTERNAL                      20U
        0 SPACE     R   2DIM ARRAY   HOLDER           21P    4D
    14300 VALMIN   *R   VARIABLE     HOLDER           18S    4D
```

```
     14301 VALRNG  *R  VARIABLE      HOLDER          19S        4D
TABLE OF EXTERNAL NAMES
-- ----------------- --

        CONREC        EXTERNAL                      28U
        DISCOL        EXTERNAL                      21U
        FLUSH         EXTERNAL                      29U        26U
        GCOLOR        EXTERNAL                      27U
        ISMAX     I   EXTERNAL                      11U
        ISMIN     I   EXTERNAL                      13U
        SETCLR        EXTERNAL                      20U
     ABBREVIATIONS USED ABOVE (THESE ARE KEYED TO THE SOURCE LISTING LINE NUMBER)
     ------------- ---- -----

     A  USED IN FORTRAN  ASSIGN STATEMENT      P  USED IN CALL/FUNC CALL OR ARRAY DEF
     D  DEFINED IN  DECLARATIVE STATEMENT      R  FORMAT  USED  IN A  READ  STATEMENT
     E  STATEMENT NUMBER ENDING A DO LOOP      S  STORED  SO CONTENTS  MAY BE CHANGED
     I  INDEX OF A DO  OR IMPLIED DO LOOP      U  NAME  USED IN EXECUTABLE  STATEMENT
     J  STATEMENT NUMBER USED IN TRANSFER      W  FORMAT  USED IN A  WRITE  STATEMENT
     L  SOURCE LINE OF A STATEMENT NUMBER      *  DEFINED  OR DECLARED  BUT NOT  USED
     N  NAME USED AS A  DO LOOP PARAMETER      ?  TEN OR  MORE REFERENCES  TO  SYMBOL
TABLE OF PARAMETERS ENCOUNTERED
--- -- ---------- -----------

 NDM1      = 72                                                  NDM2      = 88
 NR        = 5

TABLE OF LOOPS ENCOUNTERED
-- ---------- -----------

 LABEL INDEX      FROM -  TO    ADDRESS - LENGTH

    10 J            10    15        50c        27
    30 IR           23    30       132d        27
 BLOCK NAMES AND LENGTHS IN OCTAL

     164-COLCON             27-#IR              37 #CL           . 14302-HOLDER            20-GRFCOM
STATIC SPACE (IN OCTAL)
------ ----- -- -----

B SAVE:                23
I SAVE:                 4
CONSTANTS:             10
VARIABLES:             20
TEMPORARIES:           37
CODE:                 134
                   ------
TOTAL:                252
```

```
1821
1822     1.          subroutine discol (arrin,mdim,cnhold,kh,kv)
1823     2.          real arrin(mdim,kv),cnhold(kh,kv)
1824     3.          common work(5000)
1825          c
1826          c  ***********************************************************
1827          c  *   this subroutine uses DISSPLA graphics to draw contour     *
1828          c  *  plots in color.  The actual color calls are in the         *
1829          c  *  user-defined subroutine MYCON.                             *
1830          c  ***********************************************************
1831          c
1832          c     store the portion of the array to be plotted
1833     4.          do 10 i=1,kh
1834     5.          do 10 j=1,kv
1835     6.          cnhold(i,j)=arrin(i,j)
1836     7.  10      continue
1837          c     set up the contour lines
1838     8.          call bcomon (5000)
1839     9.          call conmak (cnhold,kh,kv,'scale')
1840          c     specify the characteristics of the lines and labels, including their
1841          c     color.  Color and other characteristics is set in the routine mycon.
1842    10.          call contin (0,'MYCON','LABELS',1,3)
1843    11.          call conang (60.)
1844    12.          call conthn (0.06)
1845          c     draw the contour lines
1846    13.          call contur (1,'labels','draw')
1847    14.          return
1848    15.          end
DISCOL          VECTOR LOOP BEGINS AT SEQ. NO.          5, P=      36d
```

```
TABLE OF STATEMENT NUMBERS (ALL ADDRESSES IN TABLES ARE IN OCTAL)
-- ------- --------- ----
  NUMBER USE    SOURCE PROGRAM REFERENCES

    10 UNDEF**      7I        5F       4F
    10A36d          4I
    10B63c          4I
    10C52b          5L
    10D60d          5L
 (SN=STATEMENT NUMBER, GSN=GENERATED STATEMENT NUMBER)
 (FN=FORMAT NUMBER, UNDEF**=UNDEFINED STATEMENT NUMBER)
TABLE OF NAMES ENCOUNTERED (ADDRESS FOR DUMMY ARGUMENT IS THE ARGUMENT NUMBER)
--- ------- -----------  -- -

 ADDRESS NAME     TYPE MAIN USAGE  BLOCK        SOURCE PROGRAM REFERENCES

       1 ARRIN     R   2DIM ARRAY   DUM.ARG.        6U       2D       1D
         BCOMON        EXTERNAL                     8U
       3 CNHOLD    R   2DIM ARRAY   DUM.ARG.        9P       6S       2P       1D
         CONANG        EXTERNAL                    11U
         CONLIN        EXTERNAL                    10U
         CONMAK        EXTERNAL                     9U
         CONTHN        EXTERNAL                    12U
         CONTUR        EXTERNAL                    13U
      25 DISCOL        ENTRY                        1D/2
      22 I         I   VARIABLE                     6U/2     4I
      23 J         I   VARIABLE                     6U/2     5I
       4 KH        I   VARIABLE     DUM.ARG.        9P       4N       2P       1D
       5 KV        I   VARIABLE     DUM.ARG.        9P       5N       2P/2     1D
         MAIN.         ENTRY
TABLE OF EXTERNAL NAMES
-- - -------- ----- - -

         BCOMON        EXTERNAL                     8U
         CONANG        EXTERNAL                    11U
         CONLIN        EXTERNAL                    10U
         CONMAK        EXTERNAL                     9U
         CONTHN        EXTERNAL                    12U
         CONTUR        EXTERNAL                    13U
    ABBREVIATIONS USED ABOVE (THESE ARE KEYED TO THE SOURCE LISTING LINE NUMBER)
    ------------- - -------
    A  USED IN FORTRAN  ASSIGN STATEMENT      P  USED IN CALL/FUNC CALL OR ARRAY DEF
    D  DEFINED IN  DECLARATIVE STATEMENT      R  FORMAT  USED  IN A  READ  STATEMENT
    E  STATEMENT NUMBER ENDING A DO LOOP      S  STORED  SO CONTENTS  MAY BE CHANGED
    I  INDEX OF A DO  OR IMPLIED DO LOOP      U  NAME  USED IN EXECUTABLE  STATEMENT
    J  STATEMENT NUMBER USED IN TRANSFER      W  FORMAT  USED IN A  WRITE  STATEMENT
    L  SOURCE LINE OF A STATEMENT NUMBER      *  DEFINED  OR DECLARED  BUT NOT  USED
    N  NAME USED AS A  DO LOOP PARAMETER      ?  TEN OR  MORE REFERENCES  TO  SYMBOL
```

TABLE OF LOOPS ENCOUNTERED

LABEL	INDEX	FROM -	TO	ADDRESS -	LENGTH
10	I	4	7	36d	26
10	J	5	7	52b	6

BLOCK NAMES AND LENGTHS IN OCTAL

121-DISCOL	31-#IB	25-#CL	11610-//

STATIC SPACE (IN OCTAL)

B SAVE:	27
T SAVE:	2
CONSTANTS:	20
VARIABLES:	4
TEMPORARIES:	25
CODE	75
TOTAL:	177

```
  1.          subroutine mycon (raray,iaray)
  2.          parameter (ndm1=72, ndm2=88, nr=5)
  3.          real raray(2),hue(nr)
  4.          integer iaray(9)
  5.          common /holder/ space(ndm1,ndm2),valmin,valrng,
         +      space2(ndm1,ndm2)
  6.          data hue /3hred,4hblue,5hgreen,6hyellow,5hwhite/
       c
       c  ****************************************************************
       c  *  This is the user-defined routine MYCON which sets up the line   *
       c  *  color for each contour line, using DISSPLA graphics calls.      *
       c  ****************************************************************
       c
  7.          do 10 i=1,nr
       c     set color according to level of contour value
  8.          if ((raray(i) .gt. valmin+float(nr-1)*valrng) .and.
         1        (raray(i) .lt. valmin+float(nr)*valrng))
         2        call setclr (hue(i))
  9.   10     continue
 10.          return
 11.          end
```

```
 TABLE OF STATEMENT NUMBERS (ALL ADDRESSES IN TABLES ARE IN OCTAL)
 --  -  ----------
   NUMBER USE    SOURCE PROGRAM REFERENCES

     10 UNDEF**       9I        7E
     10A23U           7I
     10BUNDEF**       7I
  00002 33b           8L
  (SN-STATEMENT NUMBER, GSN-GENERATED STATEMENT NUMBER)
   (FN-FORMAT NUMBER, UNDEF*-UNDEFINED STATEMENT NUMBER)
 TABLE OF NAMES ENCOUNTERED (ADDRESS FOR DUMMY ARGUMENT IS THE ARGUMENT NUMBER)
 ------------------- -----

  ADDRESS NAME     TYPE MAIN USAGE  BLOCK         SOURCE PROGRAM REFERENCES

          FLOAT     R   INLINE FUNCT.                8U/2
        3 HUE       R   1DIM ARRAY                   8P       6S       3D
       10 I         I   VARIABLE                     8U/3     7I
          MAIN.         ENTRY
       12 MYCON         ENTRY                        10/2
          NDM1      I   PARAMETER                    5P/2     2S
          NDM2      I   PARAMETER                    5P/2     2S
          NR        I   PARAMETER                    8P/2     7N       3P       2S
        1 RARAY     R   1DIM ARRAY   DUM.ARG.        8U/2     3D       1D
          SETCLR        EXTERNAL                     8U
    14300 VALMIN    R   VARIABLE     HOLDER          8U/2     5D
    14301 VALRNG    R   VARIABLE     HOLDER          8U/2     5D
 TABLE OF EXTERNAL NAMES
 ---------------------- -------

          SETCLR        EXTERNAL                     8U
        ABBREVIATIONS USED ABOVE (THESE ARE KEYED TO THE SOURCE LISTING LINE NUMBER)
        - ----------  --   -
        A  USED IN FORTRAN  ASSIGN STATEMENT       P  USED IN CALL/FUNC CALL OR ARRAY DEF
        D  DEFINED IN  DECLARATIVE STATEMENT       R  FORMAT  USED  IN A  READ  STATEMENT
        E  STATEMENT NUMBER ENDING A DO LOOP       S  STORED  SO CONTENTS  MAY BE CHANGED
        I  INDEX OF A DO  OR IMPLIED DO LOOP       U  NAME  USED IN EXECUTABLE  STATEMENT
        J  STATEMENT NUMBER USED IN TRANSFER       W  FORMAT  USED IN A  WRITE  STATEMENT
        L  SOURCE LINE OF A STATEMENT NUMBER       *  DEFINED  OR DECLARED  BUT NOT  USED
        N  NAME USED AS A  DO LOOP PARAMETER       ?  TEN OR  MORE REFERENCES  TO  SYMBOL
 TABLE OF PARAMETERS ENCOUNTERED
 ---- -------------  -  -- ----

  NDM1       = 72                                                  NDM2      = 88
  NR         = 5
```

TABLE OF LOOPS ENCOUNTERED

LABEL	INDEX	FROM	TO	ADDRESS	LENGTH
10	I	7	9	23d	13

BLOCK NAMES AND LENGTHS IN OCTAL

41 MYCON	15 #TB	2 #CL	30602 HOLDER

STATIC SPACE (IN OCTAL)

B SAVE	13
T SAVE:	2
CONSTANTS:	1
VARIABLES:	10
TEMPORARIES:	2
CODE:	30
TOTAL:	60

```
1871
1872     1.            subroutine adjust (tim,array,n,tmax,length)
1873          c-----------------------------------------------------------
1874          c     chops away portions of time history that are not of interest.
1875          c-----------------------------------------------------------
1876     2.            real tim(n), array(n)
1877          c
1878     3.            length=n
1879     4.            tmax=tim(n)
1880     5.            amax=array(ismax(n,array,1))
1881     6.            amin=array(1)+(amax-array(1))*0.1
1882          c
1883          c     chop to the right.
1884     7.            imax=n
1885     8.            do 10 i=n,1,-1
1886     9.            if (array(i).gt.amin) go to 20
1887    10.     10     continue
1888    11.            go to 30
1889    12.     20     imax=i
1890    13.     30     tmax=tim(imax)
1891          c
1892          c     chop to the left.
1893    14.            imin=1
1894    15.            do 40 i=1,imax
1895    16.            if (array(i).gt.amin) go to 50
1896    17.     40     continue
1897    18.            go to 60
1898    19.     50     imin=i
1899    20.     60     length=imax-imin+1
1900    21.            do 70 i=imin,imax
1901    22.            tim(i-imin+1)=tim(i)
1902    23.     70     array(i-imin+1)=array(i)
1903    24.            return
1904    25.         end
O   AT SEQUENCE NUMBER        22.
 PRNAME ADJUST    COMMENT -      DEPENDENCY INVOLVING ARRAY "TIM" ************************************************************ P=0250024C
 EXPLANATION:  AMBIGUOUS OR CONFLICTING SUBSCRIPTS
O   AT SEQUENCE NUMBER        23.
 PRNAME ADJUST    COMMENT        DEPENDENCY INVOLVING ARRAY "ARRAY" ********************************************************** P=0250024C
 EXPLANATION:  AMBIGUOUS OR CONFLICTING SUBSCRIPTS
 ADJUST                VECTOR LOOP BEGINS AT SEQ. NO.           21, P=      65d
 ADJUST   CONDITIONAL VECTOR LOOP BEGINS AT SEQ. NO.           21, P=      65d
```

```
TABLE OF STATEMENT NUMBERS (ALL ADDRESSES IN TABLES ARE IN OCTAL)
--- -- --------- -------

  NUMBER USE    SOURCE PROGRAM REFERENCES

    10 UNDEF**     10L       8F
    10A42c          8L
    10B47c          8L
    20 50a         12L       9J
    30 51a         13L      11J
    40 UNDEF**     17L      15E
    40A57b         15L
    40B64b         15L
    50 64d         19L      16J
    60 65d         20L      18J
    70 UNDEF**     23L      21I
    70A105d        21L
    70B133c        21L
 00002 116a        23F
 00003 133c        23E
 00004 UNDEF**     23E
 00005 124d        23E
  (SN=STATEMENT NUMBER, GSN=GENERATED STATEMENT NUMBER)
  (FN=FORMAT NUMBER, UNDEF**UNDEFINED STATEMENT NUMBER)
TABLE OF NAMES ENCOUNTERED (ADDRESS FOR DUMMY ARGUMENT IS THE ARGUMENT NUMBER)
--------- ------- -----

 ADDRESS NAME     TYPE MAIN USAGE   BLOCK         SOURCE PROGRAM REFERENCES

     13 ADJUST        ENTRY                       1D/2
      5 AMAX      R   VARIABLE                    6U      5S
      6 AMIN      R   VARIABLE                    16U     9U      6S
      2 ARRAY     R   1DIM ARRAY   DUM.ARG.       23U     23S     16U     9U      6U/2    5P      5U      2P      1D
     10 I         I   VARIABLE                    23U/2   22U/2   21I     19U     16U     15I     12U     9U      8I
      7 IMAX      I   VARIABLE                    21N     20U     15N     13U     12S     7S
     11 IMIN      I   VARIABLE                    23U     22U     21N     20U     19S     14S
        ISMAX     I   EXTERNAL                    5U
      5 LENGTH   *I   VARIABLE     DUM.ARG.       20S     3S      1D
        MAIN.         ENTRY
      3 N         I   VARIABLE     DUM.ARG.       8N      7U      5P      4U      3U      2P/2    1D
      1 TIM       R   1DIM ARRAY   DUM.ARG.       22U     22S     13U     4U      2D      1D
      4 TMAX     *R   VARIABLE     DUM.ARG.       13S     4S      1D
TABLE OF EXTERNAL NAMES
--- -- -------- -----

        ISMAX     I   EXTERNAL                    5U
```

```
       ABBREVIATIONS USED ABOVE (THESE ARE KEYED TO THE SOURCE LISTING LINE NUMBER)
       --- ------     --
       A  USED IN FORTRAN  ASSIGN STATEMENT      P  USED IN CALL/FUNC CALL OR ARRAY DEF
       D  DEFINED IN  DECLARATIVE STATEMENT      R  FORMAT  USED  IN A  READ  STATEMENT
       F  STATEMENT NUMBER ENDING A DO LOOP      S  STORED  SO CONTENTS  MAY BE CHANGED
       I  INDEX OF A DO  OR IMPLIED DO LOOP      U  NAME  USED IN EXECUTABLE  STATEMENT
       J  STATEMENT NUMBER USED IN TRANSFER      W  FORMAT  USED IN A  WRITE  STATEMENT
       L  SOURCE LINE OF A STATEMENT NUMBER      *  DEFINED  OR DECLARED  BUT NOT  USED
       N  NAME USED AS A  DO LOOP PARAMETER      ?  TEN OR  MORE REFERENCES  TO  SYMBOL

 TABLE OF LOOPS ENCOUNTERED
  - ---- ------- --
  LABEL INDEX      FROM -  TO   ADDRESS - LENGTH

     10 I             8    10       42c        6
     40 I            15    17       57b        5
     70 I            21    23      105d       10
  BLOCK NAMES AND LENGTHS IN OCTAL

     136 ADJUST           40 #TB             4-#CT
 STATIC SPACE (IN OCTAL)
 --------- --- ---- - --
 B SAVE:                 36
 T SAVE:                  2
 CONSTANTS:               3
 VARIABLES:               7
 TEMPORARIES:             4
 CODE:                  124
                   --------
 TOTAL:                 202
```

```
  1.            subroutine conint (arrin,xin,yin,nx,lx,ny,arrout,mx,my)
       c -----------------------------------------------------------
       c   this subroutine interplotates values from an irregularly spaced
       c   mesh to an evenly spaced mesh in two dimensions.
       c   it is meant to be an inteface to the NCAR contour plotter
       c   CONREC which works only on regularly spaced grids.
       c -----------------------------------------------------------
  2.            parameter (mxy=90)
  3.            real arrin(nx,ny), arrout(nx,my)
  4.            real xin(lx), yin(ny)
  5.            common /bound/ xmin, xmax, ymin, ymax
       c
  6.            integer iin(mxy)
  7.            real xfacm(mxy), xfacp(mxy)
       c
  8.            dy=(ymax-ymin)/float(my-1)
  9.            dx=(xmax-xmin)/float(mx-1)
 10.            nym1=ny-1
 11.            lxm1=lx-1
       c
       c  find the interval in the xin array in which x resides
       c      and find the scale factors for the x direction
       c
 12.            imin=1
 13.            do 30 iout=1,mx
 14.              x=xmin+(iout-1)*dx
 15.              if (iout.gt.1) imin=iin(iout-1)
 16.              do 10 i=imin,lxm1
 17.                if (x.gt.xin(i)) go to 10
 18.                  iin(iout)=max0(1,i-1)
 19.                  go to 20
 20.    10        continue
 21.              iin(iout)=lxm1
 22.    20        in=iin(iout)
 23.              xfacm(iout)=(x-xin(in))/(xin(in+1)-xin(in))
 24.              xfacp(iout)=1.0-xfacm(iout)
 25.    30      continue
       c
       c   cycle over the y values
       c
 26.            jin=1
 27.            do 70 jout=1,my
 28.              y=ymin+(jout-1)*dy
       c
       c   find the interval in the yin array in which y resides
       c
 29.              jmin=jin
 30.              do 40 j=jmin,nym1
 31.                if (y.gt.yin(j)) go to 40
 32.                  jin=max0(1,j-1)
 33.                  go to 50
 34.    40        continue
 35               jin=nym1
 36.    50        continue
       c
       c  compute  scale factors for y direction
```

```
   1961              c
   1962      37.               yfacm (v-yin(jin))/(yin(jin+1)-yin(jin))
   1963      38.               yfacp 1.0-yfacm
   1964              c
   1965              c   cycle over the x values
   1966              c
   1967      39.               do 60 iout=1,mx
   1968      40.                 in=rin(iout)
   1969      41.                 arrout(iout,jout)=yfacm*xfacm(iout)*arrin(in+1,jin+1)
   1970                    1     +yfacm*xfacp(iout)*arrin(in,jin+1)+yfacp*xfacm(iout)*arrin
   1971                    2     (in+1,jin)+yfacp*xfacp(iout)*arrin(in,jin)
   1972      42.          60   continue
   1973      43.          70  continue
   1974              c
   1975      44.              return
   1976      45           end
 CONINI               VECTOR LOOP BEGINS AT SEQ. NO.          39, P=      552b
```

```
 TABLE OF STATEMENT NUMBERS (ALL ADDRESSES IN TABLES ARE IN OCTAL)
 ------------ --- ------
   NUMBER USE    SOURCE PROGRAM REFERENCES

      10 510b      20L      17J      16F
      10A502d      16L
      10B513a      16L
      20 514c      22L      19J
      30 UNDEF**   25L      13E
      30A470b      13L
      30B525c      13L
      40 546c      34L      31J      30E
      40A541b      30L
      40B551b      30L
      50 552b      36L      33J
      60 UNDEF**   42L      39E
      60A573d      39L
      60B636a      39L
      70 UNDEF**   43L      27E
      70A531a      27L
      70B640d      27L
   00002 476b      15L
   00003 606a      42E
   00004 605c      42E
   00005 616a      42E
   00006 615c      42E
   00007 622a      42E
   00008 621c      42E
   00009 626c      42E
   00010 626a      42E
    (SN=STATEMENT NUMBER, GSN=GENERATED STATEMENT NUMBER)
    (FN=FORMAT NUMBER, UNDEF**=UNDEFINED STATEMENT NUMBER)
 TABLE OF NAMES ENCOUNTERED (ADDRESS FOR DUMMY ARGUMENT IS THE ARGUMENT NUMBER)
 ------------------- -- ---

   ADDRESS NAME      TYPE MAIN USAGE   BLOCK        SOURCE PROGRAM REFERENCES

         1 ARRIN     R   2DIM ARRAY   DUM ARG.       41U/4     3D       1D
         7 ARROUT   *R   2DIM ARRAY   DUM.ARG.       41S       3P       1D
       442 CONINT        ENTRY                        1D/2
       422 DX        R   VARIABLE                    14U       9S
       421 DY        R   VARIABLE                    28U       8S
           FLOAT     R   INLINE FUNCT                 9U       8U
       430 I         I   VARIABLE                    18P      17U      16I
         3 IIN       I   1DIM ARRAY                  40U      22U      21S      18S      15U      6D
       425 IMIN      I   VARIABLE                    16N      15S      12S
       431 IN        I   VARIABLE                    41U/4    40S      23U/3    22S
       426 IOUT      I   VARIABLE                    41U/5    40U      39I      24U/2    23U      22U      21U      18U      15U/2
                                                     14U      13I
       436 J         I   VARIABLE                    32P      31U      30I
       432 JIN       I   VARIABLE                    41U/4    37U/3    35S      32S      29U      26S
       435 JMIN      I   VARIABLE                    30N      29S
```

```
    433 JOUT      I   VARIABLE                     41U     28U     27I
      5 LX        I   VARIABLE      DUM.ARG.       11U      4P      1D
    424 LXM1      I   VARIABLE                     21U     16N     11S
        MAIN.         ENTRY
        MAXO      I   INLINE FUNCT                 32U     18U
     10 MX        I   VARIABLE      DUM.ARG.       39N     13N      9P      1D
        MXY       I   PARAMETER                    7P/2     6P      2S
     11 MY        I   VARIABLE      DUM.ARG        27N      8P      3P      1D
      6 NY        I   VARIABLE      DUM.ARG.       10U      4P      3P      1D
    423 NYM1      I   VARIABLE                     35U     30N     10S
    427 X         R   VARIABLE                     23U     17U     14S
    135 XFACM     R   1DIM ARRAY                   41U/2   24U     23S      7D
    267 XFACP     R   1DIM ARRAY                   41U/2   24S      7D
      2 XIN       R   1DIM ARRAY    DUM.ARG.       23U/3   17U      4D      1D
      1 XMAX      R   VARIABLE      BOUND           9U      5D     *
      0 XMIN      R   VARIABLE      BOUND          14U      9U      5D
    434 Y         R   VARIABLE                     37U     31U     28S
    437 YFACM     R   VARIABLE                     41U/2   38U     37S
    440 YFACP     R   VARIABLE                     41U/2   38S
      3 YIN       R   1DIM ARRAY    DUM.ARG.       37U/3   31U      4P      1D
      3 YMAX      R   VARIABLE      BOUND           8U      5D
      2 YMIN      R   VARIABLE      BOUND          28U      8U      5D
TABLE OF EXTERNAL NAMES
---------- - -  ---

       ABBREVIATIONS USED ABOVE (THESE ARE KEYED TO THE SOURCE LISTING LINE NUMBER)
       ------- --  -  --------
       A  USED IN FORTRAN  ASSIGN STATEMENT      P  USED IN CALL/FUNC CALL OR ARRAY DEF
       D  DEFINED IN  DECLARATIVE STATEMENT      R  FORMAT  USED  IN A  READ  STATEMENT
       E  STATEMENT NUMBER ENDING A DO LOOP      S  STORED  SO CONTENTS  MAY BE CHANGED
       I  INDEX OF A DO  OR IMPLIED DO LOOP      U  NAME  USED IN EXECUTABLE  STATEMENT
       J  STATEMENT NUMBER USED IN TRANSFER      W  FORMAT  USED IN A  WRITE  STATEMENT
       L  SOURCE LINE OF A STATEMENT NUMBER      *  DEFINED  OR DECLARED  BUT NOT  USED
       N  NAME USED AS A  DO LOOP PARAMETER      ?  TEN OR  MORE REFERENCES  TO  SYMBOL
TABLE OF PARAMETERS ENCOUNTERED
------------------  --------

 MXY      = 90

TABLE OF LOOPS ENCOUNTERED
-------------------- ---
 LABEL INDEX      FROM    TO    ADDRESS - LENGTH

    30 IOUT         13    25     470b       36
    10 I            16    20     502d       11
    70 JOUT         27    43     531a      110
    40 J            30    34     541b       10
    60 IOUT         39    42     573d       43
```

```
 BLOCK NAMES AND LENGTHS IN OCTAL

     644-CONINT          45 #IB          0 #CI          4 ROUND
 STATIC SPACE (IN OCTAL)
 --- -- - ------
 B SAVE:             40
 T SAVE:              5
 CONSTANTS:           1
 VARIABLES:         440
 TEMPORARIES:         0
 CODE:              203
                   ---
 TOTAL:             711
```

```
  1.           subroutine wsize (xmin,xmax,ymin,ymax,xlo,xhi,ylo,yhi)
        c-----------------------------------------------------------------
        c     sets plot window size.
        c-----------------------------------------------------------------
  2.           data fmin /0.1/, fmax /0.9/, fav /0.5/
        c
  3.           delx=xmax-xmin
  4.           dely=ymax-ymin
  5.           if (dely.gt.delx) go to 10
  6.           xlo=fmin
  7.           xhi=fmax
  8.           fy=(xhi-xlo)*dely/delx
  9.           yhi=fav+0.5*fy
 10.           ylo=yhi-fy
 11.           return
 12.     10    ylo=fmin
 13.           yhi=fmax
 14.           fx=(yhi-ylo)*delx/dely
 15.           xhi=fav+0.5*fx
 16.           xlo=xhi-fx
 17            return
 18.       end
```

```
TABLE OF STATEMENT NUMBERS (ALL ADDRESSES IN TABLES ARE IN OCTAL)
-------------------------

  NUMBER USE    SOURCE PROGRAM REFERENCES

    10 36d         12I        5J
   (SN=STATEMENT NUMBER, GSN=GENERATED STATEMENT NUMBER)
   (FN=FORMAT NUMBER, UNDEF*=UNDEFINED STATEMENT NUMBER)
TABLE OF NAMES ENCOUNTERED (ADDRESS FOR DUMMY ARGUMENT IS THE ARGUMENT NUMBER)
---------------------------

 ADDRESS NAME     TYPE MAIN USAGE   BLOCK        SOURCE PROGRAM REFERENCES

       6 DELX      R   VARIABLE                   14U      8U      5U      3S
       7 DELY      R   VARIABLE                   14U      8U      5U      4S
       5 FAV       R   VARIABLE                   15U      9U      2S
       4 FMAX      R   VARIABLE                   13U      7U      2S
       3 FMIN      R   VARIABLE                   12U      6U      2S
      11 FX        R   VARIABLE                   16U     15U     14S
      10 FY        R   VARIABLE                   10U      9U      8S
         MAIN.         ENTRY
      13 WSIZE         ENTRY                      10/2
       6 XHI       R   VARIABLE      DUM.ARG.     16U     15S      8U      7S      1D
       5 XLO       R   VARIABLE      DUM.ARG.     16S      8U      6S      1D
       2 XMAX      R   VARIABLE      DUM.ARG.      3U      1D
       1 XMIN      R   VARIABLE      DUM.ARG.      3U      1D
      10 YHI       R   VARIABLE      DUM.ARG.     14U     13S     10U      9S      1D
       7 YLO       R   VARIABLE      DUM.ARG.     14U     12S     10S      1D
       4 YMAX      R   VARIABLE      DUM.ARG.      4U      1D
       3 YMIN      R   VARIABLE      DUM.ARG.      4U      1D
TABLE OF EXTERNAL NAMES
-----------------

       ABBREVIATIONS USED ABOVE (THESE ARE KEYED TO THE SOURCE LISTING LINE NUMBER)
       -------------

       A  USED IN FORTRAN  ASSIGN STATEMENT     P  USED IN CALL/FUNC CALL OR ARRAY DEF
       D  DEFINED IN  DECLARATIVE STATEMENT     R  FORMAT  USED  IN A  READ  STATEMENT
       E  STATEMENT NUMBER ENDING A DO LOOP     S  STORED  SO CONTENTS  MAY BE CHANGED
       I  INDEX OF A DO  OR IMPLIED DO LOOP     U  NAME  USED IN EXECUTABLE  STATEMENT
       J  STATEMENT NUMBER USED IN TRANSFER     W  FORMAT  USED IN A  WRITE  STATEMENT
       L  SOURCE LINE OF A STATEMENT NUMBER     *  DEFINED  OR DECLARED  BUT NOT  USED
       N  NAME USED AS A  DO LOOP PARAMETER     ?  TEN OR  MORE REFERENCES  TO  SYMBOL

 BLOCK NAMES AND LENGTHS IN OCTAL

     52-WSIZE            14 #IB               0 #CI
STATIC SPACE (IN OCTAL)
-----------------
```

```
B SAVE:            14
T SAVE:             0
CONSTANTS:          1
VARIABLES:         11
TEMPORARIES:        0
CODE:              40

TOTAL:             66
```

```
 1.        subroutine eospl (rrr,eee,n,gamma,ppp)
      c---------------------------------------------------------------
      c     equation of state routine for air.
      c
      c     input varible definitions.
      c          rrr = mass density
      c          eee = internal energy per unit volume
      c                (converted for internal use to energy per unit mass)
      c            n = number of entries in arrays rrr & eee
      c
      c     this routine calculates gamma and ppp as follows:
      c        gamma = 1.0 + p/(rho*e)
      c        ppp   = pressure(units of eee)
      c---------------------------------------------------------------
 2.        parameter (m=92)
      c
 3.     dimension rrr(n), eee(n), gamma(n), ppp(n)
 4.     dimension t11(m), t12(m), t21(m), t22(m), rho(m), e(m), int(m)
 5.     dimension p(m), omp(m), q(m), i(m), j(m), tem(m)
 6.     dimension g(8,105), jcy(m), js(m)
 7.     dimension g1(120), g2(120), g3(120), g4(120), g5(120), g6(120)
 8.     dimension g7(120), gf(840)
      c
 9.     equivalence (g1(1),g(1,1)), (g2(1),g(1,16)), (g3(1),g(1,31))
10.     equivalence (g4(1),g(1,46)), (g5(1),g(1,61)), (g6(1),g(1,76))
11.     equivalence (g7(1),g(1,91)), (int(1),tem(1)), (gf(1),g(1,1))
      c
12.        data rzeroi /774.413/
13.     data xl16e /2.772588722239774483568908181041479110717773437 5/
14.     data rl16e /0.360673760222240957773465197533369064331054687 5/
      c---------------------------------------------------------------
      c     q = gamma - 1.0 is stored for 32 bit word machines in powers of
      c     16 across for mass density variation and intermediate values
      c     1 - 16 for powers of 16 verticaly which represent the internal
      c     energy variation.
      c
      c     16**(2) .ge. rho .ge. 16**(-6)
      c     16**(8) .le. e   .le. 16**(15)
      c---------------------------------------------------------------
15.     data g1 /8*.4222,8*.4152,8*.4110,8*.4081,8*.4058,8*.4040,8*.4024,8
       1 *.4011,8*.3998,8*.3988,8*.3978,8*.3969,8*.3961,8*.3953,8*.3935/
16.     data g2 /.3918,.3918,.3918,.3918,.3918,.3918,.3918,.3918,.3723,.37
       1 15,.3707,.3699,.3690,.3680,.3663,.3637,.3555,.3538,.3522,.3502,.3
       2 476,.3430,.3344,.3238,.3370,.3370,.3370,.3364,.3347,.3277,.3099,.
       3 2885,.3257,.3227,.3201,.3134,.3062,.3014,.2884,.2591,.3166,.3110,
       4 .3063,.2946,.2831,.2783,.2677,.2358,.3111,.3006,.2940,.2787,.2635
       5 ,.2588,.2502,.2236,.3075,.2906,.2810,.2665,.2466,.2418,.2350,.213
       6 1,.3043,.2819,.2695,.2554,.2317,.2269,.2216,.2038,.2929,.2740,.25
       7 93,.2455,.2206,.2136,.2097,.1955,.2840,.2672,.2500,.2366,.2166,.2
       8 015,.1988,.1879,.2764,.2611,.2429,.2285,.2125,.1890,.1890,.1811,.
       9 2714,.2555,.2384,.2210,.2079,.1818,.1799,.1747,.2669,.2504,.2343,
       $ .2141,.2037,.1822,.1709,.1689,.2624,.2473,.2304,.2096,.1998,.1828
       $ ,.1684,.1639/
17.     data g3 /.2599,.2446,.2268,.2087,.1961,.1834,.1673,.1601,.2401,.21
       1 91,.1972,.1775,.1592,.1444,.1358,.1203,.2002,.1960,.1749,.1536,.1
       2 376,.1252,.1107,.1044,.1911,.1829,.1633,.1420,.1266,.1101,.1012,.
```

```
          3 0933,.1950,.1781,.1566,.1415,.1241,.1118,.1009,.0948,.2001,.1789,
          4 .1594,.1443,.1306,.1189,.1095,.1013,.2040,.1826,.1657,.1494,.1338
          5 ,.1177,.1081,.0980,.2034,.1854,.1683,.1497,.1322,.1169,.1051,.094
          6 6,.1969,.1855,.1685,.1487,.1304,.1149,.1024,.0916,.1899,.1837,.16
          7 77,.1475,.1287,.1126,.1002,.0900,.1841,.1817,.1667,.1464,.1272,.1
          8 109,.0983,.0888,.1800,.1800,.1659,.1455,.1262,.1097,.0965,.0878,.
          9 1779,.1787,.1657,.1450,.1254,.1087,.0949,.0868,.1773,.1778,.1656,
          $ .1447,.1250,.1080,.0939,.0859,.1783,.1778,.1658,.1448,.1248,.1076
          $ ,.0933,.0851/
 18.       data g4 /.1808,.1781,.1667,.1451,.1248,.1074,.0930,.0843,.2134,.20
          1 40,.1978,.1782,.1565,.1368,.1206,.1074,.2210,.2072,.1957,.1739,.1
          2 516,.1312,.1137,.1000,.2245,.2109,.1989,.1772,.1563,.1390,.1247,.
          3 1133,.2299,.2132,.2017,.1795,.1579,.1384,.1221,.1090,.2350,.2157,
          4 .2023,.1798,.1575,.1370,.1197,.1057,.2397,.2194,.2034,.1796,.1572
          5 ,.1372,.1205,.1070,.2452,.2227,.2050,.1805,.1576,.1379,.1236,.111
          6 8,.2510,.2256,.2069,.1814,.1581,.1383,.1231,.1103,.2560,.2282,.20
          7 91,.1822,.1585,.1385,.1226,.1083,.2605,.2312,.2111,.1829,.1588,.1
          8 386,.1222,.1070,.2677,.2358,.2129,.1836,.1592,.1386,.1218,.1071,.
          9 2759,.2403,.2145,.1857,.1598,.1389,.1219,.1078,.2834,.2445,.2160,
          $ .1878,.1603,.1394,.1223,.1084,.2905,.2484,.2175,.1898,.1613,.1399
          $ ,.1226,.1090/
 19.       data g5 /.2963,.2531,.2199,.1918,.1625,.1407,.1230,.1096,.4323,.35
          1 82,.3109,.2889,.2803,.2706,.2410,.2224,.4610,.4026,.3624,.3212,.2
          2 926,.2551,.2375,.2015,.4199,.3837,.3401,.2979,.2623,.2318,.2108,.
          3 1854,.3924,.3642,.3194,.2760,.2427,.2157,.1902,.1721,.3794,.3479,
          4 .3025,.2673,.2311,.2019,.1842,.1613,.3674,.3448,.2961,.2593,.2255
          5 ,.1994,.1785,.1594,.3573,.3443,.2910,.2517,.2293,.2006,.1843,.167
          6 9,.3661,.3438,.2935,.2597,.2336,.2225,.2143,.2116,.3674,.3435,.30
          7 80,.2728,.2606,.2577,.2573,.2573,.3685,.3453,.3210,.3014,.2942,.2
          8 933,.2932,.2932,.3814,.3612,.3341,.3276,.3257,.3253,.3252,.3252,.
          9 3903,.3752,.3570,.3522,.3513,.3510,.3506,.3496,.4012,.3899,.3782,
          $ .3751,.3743,.3741,.3734,.3713,.4155,.4057,.3956,.3930,.3920,.3913
          $ ,.3907,.3890/
 20.       data g6 /.4290,.4205,.4118,.4092,.4077,.4065,.4059,.4047,.5411,.53
          1 85,.5359,.5353,.5351,.5350,.5350,.5350,.5823,.5812,.5801,.5797,.5
          2 796,.5797,.5797,.5797,.6096,.6090,.6085,.6082,.6082,.6083,.6083,.
          3 6083,.6308,.6306,.6305,.6303,.6303,.6305,.6305,.6305,.6481,.6483,
          4 .6485,.6483,.6484,.6486,.6487,.6487,.6627,.6632,.6637,.6636,.6637
          5 ,.6640,.6640,.6640,.6754,.6761,.6769,.6768,.6770,.6773,.6773,.677
          6 3,.6866,.6875,.6885,.6884,.6886,.6890,.6890,.6890,.6966,.6977,.69
          7 89,.6989,.6991,.6995,.6995,.6995,.7056,.7070,.7083,.7083,.7085,.7
          8 090,.7090,.7090,.7139,.7154,.7169,.7169,.7172,.7176,.7177,.7177,.
          9 7214,.7231,.7248,.7248,.7251,.7256,.7256,.7256,.7285,.7303,.7321,
          $ .7321,.7325,.7330,.7330,.7330,.7350,.7370,.7390,.7390,.7393,.7398
          $ ,.7399,.7399/
 21.       data g7 /.7411,.7432,.7453,.7454,.7457,.7463,.7463,.7463,.8069,.81
          1 03,.8138,.8139,.8145,.8152,.8153,.8153,.8454,.8496,.8538,.8540,.8
          2 547,.8556,.8557,.8557,.8727,.8774,.8822,.8825,.8832,.8842,.8843,.
          3 8843,.8938,.8990,.9042,.9046,.9054,.9064,.9065,.9065,.9111,.9166,
          4 .9222,.9226,.9235,.9246,.9247,.9247,.9258,.9316,.9374,.9379,.9387
          5 ,.9399,.9400,.9400,.9384,.9445,.9506,.9511,.9520,.9532,.9533,.953
          6 3,.9496,.9559,.9622,.9627,.9637,.9649,.9650,.9650,.9596,.9661,.97
          7 27,.9731,.9741,.9754,.9755,.9755,.9686,.9753,.9821,.9826,.9836,.9
          8 849,.9850,.9850,.9769,.9837,.9906,.9912,.9922,.9936,.9937,.9937,.
          9 9845,.9915,.9986,.9991,4*.9999,.9915,.9987,6*.9999,.9981,7*.9999/
     c
```

```
c     real air eos, table lookup on gilmore data.  (no temp. model)
c     to avoid costly logrithmic functions the table "g" is stored in a
c     form so that the hexadecimal word structure of a 32 bit machine
c     may be exploited
c     this logic may be transfered to other machines by recalculating
c     the table "q" appropriate to the word arcitecture of that machine.
c     machine dependent functions and key numbers must also be changed.
c     ------------------------------------------------------------------
         ist=0
         nr=n
   10    nst=min0(nr,m)
c
         do 20 ire=1,nst
         rho(ire)=rzeroi*rrr(ist+ire)
         e(ire)=amax1(3.e8,eee(ist+ire)/rrr(ist+ire))
c
c     calculate mass density variation index "i".
         tem(ire)=alog(rho(ire))*r116e+500.0
         i(ire)=tem(ire)
         omp(ire)=tem(ire)-i(ire)
         i(ire)=502-i(ire)
         p(ire)=1.0-omp(ire)
         i(ire)=max0(i(ire),1)
c
c     calculate internal energy variation index "j".
         tem(ire)=alog(e(ire))*r116e
         jcy(ire)=ifix(tem(ire))
         tem(ire)=tem(ire)-jcy(ire)
         tem(ire)=exp(x116e*tem(ire))
         jcy(ire)=jcy(ire)-7
         js(ire)=tem(ire)
         q(ire)=tem(ire)-js(ire)
         j(ire)=js(ire)+15*jcy(ire)
         j(ire)=min0(j(ire),104)
         j(ire)=i(ire)*8+j(ire)
   20    i(ire)=j(ire)-8
         do 30 ire=1,nst
         t11(ire)=qf(i(ire))
         t21(ire)=qf(i(ire)+1)
         t12(ire)=qf(j(ire))
   30    t22(ire)=qf(j(ire)+1)
c
c     calculate gamma by linear interpolation.
         do 40 ire=1,nst
         t12(ire)=t12(ire)-t11(ire)
         t22(ire)=t22(ire)-t21(ire)
   40    gamma(ist+ire)=1.0+omp(ire)*(t11(ire)+q(ire)*t12(ire))+p(ire)*
     1   (t21(ire)+q(ire)*t22(ire))
c
         nr=nr-nst
         ist=ist+nst
         if (nr.gt.0) go to 10
c
c     calculate presure in units of eee
         do 50 ire=1,n
   50    ppp(ire)=amax1(0.0,eee(ire)*(gamma(ire)-1.0))
```

```
   2167            c
   2168    59.            return
   2169    60.         end
  EOSP1               VECTOR LOOP BEGINS AT SEQ. NO.        25, P=    4143c
  EOSP1               VECTOR LOOP BEGINS AT SEQ. NO.        50, P=    4267c
  EOSP1               VECTOR LOOP BEGINS AT SEQ. NO.        57, P=    4322d
```

```
TABLE OF STATEMENT NUMBERS (ALL ADDRESSES IN TABLES ARE IN OCTAL)

  NUMBER USE     SOURCE PROGRAM REFERENCES

    10 4143c       56U     24I
    20 UNDEF**     44L     25E
    20A4162b       25L
    20B4252b       25I
    30 UNDEF**     49L     45E
    30A4257d       45I
    30B4267c       45I
    40 UNDEF**     53L     50E
    40A4277a       50L
    40B4322d       50I
    50 UNDEF**     58L     57E
    50A4340c       57L
    50B4351b       57L
  (SN=STATEMENT NUMBER, GSN=GENERATED STATEMENT NUMBER)
  (FN=FORMAT NUMBER, UNDEF**UNDEFINED STATEMENT NUMBER)
TABLE OF NAMES ENCOUNTERED (ADDRESS FOR DUMMY ARGUMENT IS THE ARGUMENT NUMBER)

 ADDRESS NAME     TYPE MAIN USAGE   BLOCK       SOURCE PROGRAM REFERENCES

         ALOG      R   EXTERNAL                  34U     28U
         AMAX1     R   INLINE FUNCT.             58U     27U
    2564 F         R   1DIM ARRAY                34P     27S     4D
       2 EEE       R   1DIM ARRAY   DUM ARG.     58P     27P     3P      1D
    4134 EOSP1         ENTRY                      10/2
         EXP       R   EXTERNAL                  37U
       4 G1       *R   1D EQ. ARRAY               9D     15S     7D
     174 G2       *R   1D EQ. ARRAY               9D     16S     7D
     364 G3       *R   1D EQ. ARRAY               9D     17S     7D
     554 G4       *R   1D EQ. ARRAY              10D     18S     7D
     744 G5       *R   1D EQ. ARRAY              10D     19S     7D
    1134 G6       *R   1D EQ. ARRAY              10D     20S     7D
    1324 G7       *R   1D EQ. ARRAY              11D     21S     8D
       4 GAMMA     R   1DIM ARRAY   DUM.ARG.     58U     53S     3P      1D
       4 GF        R   UNDEF EQUIV               49U     48U     47U     46U     11D     8D
    3344 I         I   1DIM ARRAY                47U     46U     44S     43U     33P     33S     31U     31S     30U
                                                 29S     5D
         IFIX      I   INLINE FUNCT.             35U
    4132 IRE       I   VARIABLE                  58U/3   57I     53U/9   52U/3   51U/3   50I     49U/2   48U/2   47U/2
                                                 46U/2   45I     44U/2   43U/3   42U/2   41U/3   40U/3   39U/2   38U/2
                                                 37U/2   36U/3   35U/2   34U/2   33U/2   32U/2   31U/2   30U/3   29U/2
                                                 28U/2   27U/3   26U/2   25I
    4127 IST       I   VARIABLE                  55U     55S     53U     27U/2   26U     22S
    3500 J         I   1DIM ARRAY                49U     48U     44U     43U     43S     42P     42S     41S     5D
    3634 JCY       I   1DIM ARRAY                41U     38U     38S     36U     35S     6D
    3770 JS        I   1DIM ARRAY                41U     40U     39S     6D
         M         I   PARAMETER                 24P     6P/2    5P/6    4P/7    2S
         MAIN.         ENTRY
```

```
        MAXO      I   INLINE FUNCT                 33U
        MINO      I   INLINE FUNCT.                42U    24U
      3 N         I   VARIABLE      DUM.ARG.       57N    23U    3P/4    1D
   4130 NR        I   VARIABLE                     56U    54U    54S    24P    23S
   4131 NST       I   VARIABLE                     55U    54U    50N    45N    25N    24S
   3054 OMP       R   1DIM ARRAY                   53U    32U    30S     5D
   2720 P         R   1DIM ARRAY                   53U    32S     5D
      5 PPP      *R   1DIM ARRAY    DUM.ARG.       58S     3P     1D
   3210 Q         R   1DIM ARRAY                   53U/2  40S     5D
   2430 RHO       R   1DIM ARRAY                   28P    26S     4D
   4126 RLIGE     R   VARIABLE                     34U    28U    14S
      1 RRR       R   1DIM ARRAY    DUM.ARG.       27U    26U     3D     1D
   4124 RZEROI    R   VARIABLE                     26U    12S
   1650 T11       R   1DIM ARRAY                   53U    51U    46S     4D
   2004 T12       R   1DIM ARRAY                   53U    51U    51S    48S     4D
   2140 T21       R   1DIM ARRAY                   53U    52U    47S     4D
   2274 T22       R   1DIM ARRAY                   53U    52U    52S    49S     4D
   1514 TEM       R   1D EQ. ARRAY                 40U    39U    37U    37S    36U    36S    35P    34S    30U
                                                   29U    28S    11D     5D
   4125 XLIGE     R   VARIABLE                     37P    13S
TABLE OF EXTERNAL NAMES
------------------ ----

        ALOG      R   EXTERNAL                     34U    28U
        EXP       R   EXTERNAL                     37U
      ABBREVIATIONS USED ABOVE (THESE ARE KEYED TO THE SOURCE LISTING LINE NUMBER)
      ------------- -----------
      A  USED IN FORTRAN  ASSIGN STATEMENT      P  USED IN CALL/FUNC CALL OR ARRAY DEF
      D  DEFINED IN  DECLARATIVE STATEMENT      R  FORMAT  USED  IN A  READ  STATEMENT
      E  STATEMENT NUMBER ENDING A DO LOOP      S  STORED  SO CONTENTS  MAY BE CHANGED
      I  INDEX OF A DO  OR IMPLIED DO LOOP      U  NAME  USED IN EXECUTABLE  STATEMENT
      J  STATEMENT NUMBER USED IN TRANSFER      W  FORMAT  USED IN A  WRITE  STATEMENT
      L  SOURCE LINE OF A STATEMENT NUMBER      *  DEFINED  OR DECLARED  BUT NOT  USED
      N  NAME USED AS A  DO LOOP PARAMETER      ?  TEN OR  MORE REFERENCES  TO  SYMBOL
TABLE OF PARAMETERS ENCOUNTERED
- --------------------------

 M        = 92

TABLE OF LOOPS ENCOUNTERED
 -------- -

 LABEL INDEX      FROM -  TO   ADDRESS - LENGTH

    20 IRE          25    44    4162b      70
    30 IRE          45    49    4257d      10
    40 IRE          50    53    4277a      23
    50 IRE          57    58    4340c      11
 BLOCK NAMES AND LENGTHS IN OCTAL
```

```
     4354 EOSP1                240 #1B              3 #C1
 STATIC SPACE (IN OCTAL)
 -----------------------
 B SAVE:                    33
 T SAVE:                     5
 CONSTANTS:                  2
 VARIABLES:               4131
 TEMPORARIES:              203
 CODE:                     221
                          ----
 TOTAL:                   4617
```

```
2170    1.          subroutine eosp2 (rrr,eee,n,gamma,ppp)
2171          c ----------------------------------------------------
2172          c     equation of state for dust.
2173          c ----------------------------------------------------
2174    2.          real rrr(n), eee(n), gamma(n), ppp(n)
2175    3           data gama,gamam1 /1.4,0.4/
2176          c
2177    4           do 10 i=1,n
2178    5.          gamma(i)=gama
2179    6.     10   ppp(i)=eee(i)*gamam1
2180    7.          return
2181    8           end
EOSP2          VECTOR LOOP BEGINS AT SEQ. NO.        4, P=      13b
```

```
TABLE OF STATEMENT NUMBERS (ALL ADDRESSES IN TABLES ARE IN OCTAL)

  NUMBER USE    SOURCE PROGRAM REFERENCES

    10 UNDEF**      6L        4L
    10A27d          4L
    10B37a          4L
  (SN=STATEMENT NUMBER, GSN=GENERATED STATEMENT NUMBER)
  (FN=FORMAT NUMBER, UNDEF**=UNDEFINED STATEMENT NUMBER)
TABLE OF NAMES ENCOUNTERED (ADDRESS FOR DUMMY ARGUMENT IS THE ARGUMENT NUMBER)

 ADDRESS NAME     TYPE MAIN USAGE  BLOCK          SOURCE PROGRAM REFERENCES

      2 FEE        R   1DIM ARRAY   DUM.ARG.          6U       2P       1D
      7 EOSP2          ENTRY                          1D/2
      3 GAMA       R   VARIABLE                       5U       3S
      4 GAMAM1     R   VARIABLE                       6U       3S
      4 GAMMA     *R   1DIM ARRAY   DUM.ARG.          5S       2P       1D
      5 I          I   VARIABLE                       6U/2     5U       4L
        MAIN.          ENTRY
      3 N          I   VARIABLE     DUM.ARG.          4N       2P/4     1D
      5 PPP       *R   1DIM ARRAY   DUM.ARG.          6S       2P       1D
TABLE OF EXTERNAL NAMES

      ABBREVIATIONS USED ABOVE (THESE ARE KEYED TO THE SOURCE LISTING LINE NUMBER)

      A  USED IN FORTRAN  ASSIGN STATEMENT     P  USED IN CALL/FUNC CALL OR ARRAY DEF
      D  DEFINED IN  DECLARATIVE STATEMENT     R  FORMAT  USED  IN A  READ  STATEMENT
      E  STATEMENT NUMBER ENDING A DO LOOP     S  STORED  SO CONTENTS  MAY BE CHANGED
      I  INDEX OF A DO  OR IMPLIED DO LOOP     U  NAME  USED IN EXECUTABLE  STATEMENT
      J  STATEMENT NUMBER USED IN TRANSFER     W  FORMAT  USED IN A  WRITE  STATEMENT
      L  SOURCE LINE OF A STATEMENT NUMBER     *  DEFINED  OR DECLARED  BUT NOT  USED
      N  NAME USED AS A  DO LOOP PARAMETER     ?  TEN OR  MORE REFERENCES  TO  SYMBOL

TABLE OF LOOPS ENCOUNTERED

 LABEL INDEX       FROM -  TO    ADDRESS - LENGTH

   10 I              4     6       27d        10
 BLOCK NAMES AND LENGTHS IN OCTAL

     42-EOSP2            27 #IB              0-#CL
```

```
STATIC SPACE (IN OCTAL)
------------ ---  -----
B SAVE:             25
T SAVE:              2
CONSTANTS:           1
VARIABLES:           5
TEMPORARIES:         0
CODE:               34
                -- -----
TOTAL:              71
```

```
2198            [...]rscr(jk)
2199    c       go to (60,40,20), nspec
2200    c       species 3
2201  10.  20   call eosp3 (rho,ein,np,gscr,pscr)
2202  11.       do 30 jk=1,np
2203  12.       scr3(jk)=rhb(jk)*scr1(jk)
2204  13.       scr3(jk)=cvmgt(0.0,scr3(jk),scr3(jk).lt.0.1)
2205  14.       pre(jk)=pre(jk)+pscr(jk)*scr3(jk)
2206  15.       gam(jk)=gam(jk)+(gscr(jk)-1.0)*scr3(jk)
2207  16.  30   rscr(jk)=cvmgt(rscr(jk),rscr(jk)-rhb(jk),scr3(jk).lt.0.1)
2208    c
2209    c       species 2
2210  17.  40   call eosp2 (rho,ein,np,gscr,pscr)
2211  18.       do 50 jk=1,np
2212  19.       scr2(jk)=rha(jk)*scr1(jk)
2213  20.       scr2(jk)=cvmgt(0.0,scr2(jk),scr2(jk).lt.0.1)
2214  21.       pre(jk)=pre(jk)+pscr(jk)*scr2(jk)
2215  22.       gam(jk)=gam(jk)+(gscr(jk)-1.0)*scr2(jk)
2216  23.  50   rscr(jk)=cvmgt(rscr(jk),rscr(jk)-rha(jk),scr2(jk).lt.0.1)
2217    c
2218    c       species 1.
2219  24.  60   call eosp1 (rho,ein,np,gscr,pscr)
2220  25.       do 70 jk=1,np
2221  26.       scr1(jk)=amax1(0.0,rscr(jk)*scr1(jk))
2222  27.       pre(jk)=amax1(0.0,pre(jk)+pscr(jk)*scr1(jk))
2223  28.  70   gam(jk)=gam(jk)+(gscr(jk)-1.0)*scr1(jk)
EOS   29.       return
EOS   30.       end
EOS     VECTOR LOOP BEGINS AT SEQ. NO.    4, P=
EOS     VECTOR LOOP BEGINS AT SEQ. NO.   11, P=    13b
        VECTOR LOOP BEGINS AT SEQ. NO.   18, P=    52a
        VECTOR LOOP BEGINS AT SEQ. NO.   25, P=   120d
                                                  167c
```

2	NSPEC	I	VARIABLE											
6	PRE	R	1DIM ARRAY	DUM.ARG.										
10	PSCR	R	1DIM ARRAY	DUM.ARG.	[illegible]									
15	RHA	R	1DIM ARRAY	DUM.ARG.	23U	19U								
16	RHB	R	1DIM ARRAY	DUM.ARG.	16U	12U	3P	[illegible]						
3	RHO	R	1DIM ARRAY	DUM.ARG.	24P	17P	10P	5U	[illegible]					
7	RSCR	R	1DIM ARRAY	DUM.ARG.	26P	23P/2	23S	16P/2	16S	8U	[illegible]			
12	SCR1	R	1DIM ARRAY	DUM.ARG.	28U	27U	26U	26S	19U	12U	8S	[illegible]		
13	SCR2	R	1DIM ARRAY	DUM.ARG.	23P	22U	21U	20P/2	20S	19S	3P	1D		
14	SCR3	R	1DIM ARRAY	DUM.ARG.	16P	15U	14U	13P/2	13S	12S	3P	1D		

```
TABLE OF EXTERNAL NAMES
-------- -- -------- -----

     EOSP1     EXTERNAL                  24U
     EOSP2     EXTERNAL                  17U
     EOSP3     EXTERNAL                  10U
   ABBREVIATIONS USED ABOVE (THESE ARE KEYED TO THE SOURCE LISTING LINE NUMBER)
   ------------- ---- -----
   A  USED IN FORTRAN  ASSIGN STATEMENT     P  USED IN CALL/FUNC CALL OR ARRAY DEF
   D  DEFINED IN  DECLARATIVE STATEMENT     R  FORMAT  USED  IN A  READ  STATEMENT
   E  STATEMENT NUMBER ENDING A DO LOOP     S  STORED  SO CONTENTS  MAY BE CHANGED
   I  INDEX OF A DO  OR IMPLIED DO LOOP     U  NAME  USED IN EXECUTABLE  STATEMENT
   J  STATEMENT NUMBER USED IN TRANSFER     W  FORMAT  USED IN A  WRITE  STATEMENT
   L  SOURCE LINE OF A STATEMENT NUMBER     *  DEFINED  OR DECLARED  BUT NOT  USED
   N  NAME USED AS A  DO LOOP PARAMETER     ?  TEN OR  MORE REFERENCES  TO  SYMBOL

TABLE OF LOOPS ENCOUNTERED
-----  --  -----  -----------

 LABEL INDEX      FROM    TO   ADDRESS - LENGTH

    10 JK            4     8       30d       13
    30 JK           11    16       74c       24
    50 JK           18    23      143b       24
    70 JK           25    28      207d       22
 BLOCK NAMES AND LENGTHS IN OCTAL

    234-FDS               60-#1B              22-#CL
STATIC SPACE (IN OCTAL)
------------------- --
B SAVE:                57
T SAVE:                 1
CONSTANTS:              2
VARIABLES:              3
TEMPORARIES:           22
CODE:                 227
                 --------
TOTAL:                336
```

INITIAL PAGES OF PROGRAM UNITS

```
      ADJUST        88
      COLCON        79
      CONINT        91
      CONV3D        13
      DISCOL        82
      DISCON        43
      DISPRF        37
      DISTPL        40
      EOS          109
      FOSP1         99
      FOSP2        106
      FILEIO        61
      FLIK3D         1
      MYCON         85
      PLTARR        65
      PROF3D        46
      RESTRT        72
      WSIZE         96
***  COMPILATION SUMMARY  ***

EXECUTE LINE  -  I=nplot3d on=adix
CPU SECONDS   -    3.1893
I/O SECONDS   -    1.0630
SYS SECONDS   -    0.0472
SIZE(OCTAL)   -    601000
```

END
DATE
FILMED
DTIC
JULY 88